LINUX 101 HACKS

Practical Examples to Build a Strong Foundation in Linux

Ramesh Natarajan
www.thegeekstuff.com

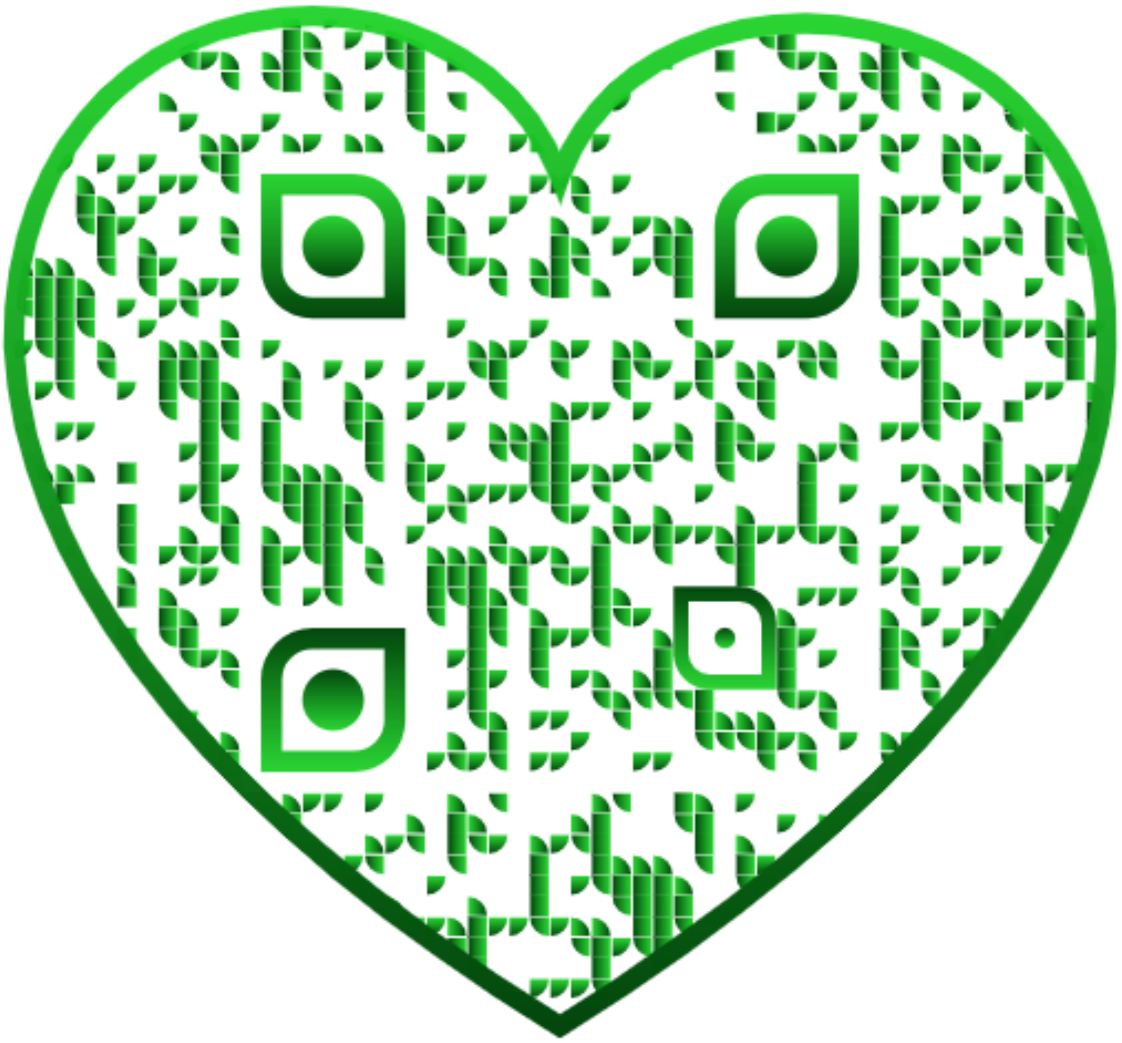


Table of Contents

Introduction....7

About the Author....8

Copyright & Disclaimer....9

Version....9

More eBooks from The Geek Stuff....10

Bash 101 Hacks....10

Sed and Awk 101 Hacks....11

Nagios Core....12

Vim 101 Hacks....13

Chapter 1: Powerful CD Command Hacks....14

Hack 1. Define CD Base Directory Using CDPATH 14

Hack 2. Use CD Alias to Navigate Up the Directory....15

Hack 3. Perform mkdir and cd Using a Single Command 18

Hack 4. Toggle Between Directories 19

Hack 5. Manipulate Directory Stack....20

Hack 6. Automatically Correct Mistyped Directory Names . 23

Chapter 2: Essential Linux Commands 24

Hack 7. Grep Command 24

Hack 8. Regular Expression in Grep....26

Hack 9. Find Command 29

Hack 10. Suppress Standard Output and Error Message 31

Hack 11. Join Command 32

Hack 12. Change the Case 33

Hack 13. Xargs Command 34

Hack 14. Sort Command 35

Hack 15. Uniq Command 37

Hack 16. Cut Command 38

Hack 17. Stat Command 39

Hack 18. Diff Command 41

Hack 19. Display Total Connect Time of Users 42
Hack 20. Execute Commands in the Background 43
Hack 21. Sed Basics - Find and Replace Using RegEx 45
Hack 22: Awk Introduction – Print Examples 50
Hack 23. Vim Editor Navigation Fundamentals 56
Hack 24. Chmod Command Examples.....60
Hack 25. View Multiple Log Files in One Terminal 62
Hack 26. Less Command.....64
Hack 27. Wget Examples 68

Chapter 3: SSH Commands and Tips.....76
Hack 28. Debug SSH Client Session 76
Hack 29. Toggle SSH Session using SSH Escape Character .77
Hack 30. Display SSH Session Statistics 78
Hack 31. Change OpenSSH Security Options.....79
Hack 32. Transfer All PuTTY Sessions.....85

Chapter 4: Date Manipulation 87
Hack 33. Set System Date and Time 87
Hack 34. Set Hardware Date and Time 88
Hack 35. Display Date and Time in a Specific Format 89
Hack 36. Display Past Date and Time 90
Hack 37. Display Future Date and Time 91

Chapter 5: PS1, PS2, PS3, PS4 and PROMPT_COMMAND.....93
Hack 38. PS1 - Default Interaction Prompt 93
Hack 39. PS2 - Continuation Interactive Prompt 94
Hack 40. PS3 - Prompt Used by Select Command.....95
Hack 41. PS4 - Prompt to Prefix Tracing Output 97
Hack 42. PROMPT_COMMAND 98
Hack 43. Customize Bash Prompt Using PS1.....99
Hack 44. Colorful Bash Prompt Using PS1.....104

Chapter 6: Archive and Compression.....109
Hack 45. Zip Command Basics 109
Hack 46. Zip Command Advanced Compression.....111

Hack 47. Password Protection of Zip files 113
Hack 48. Tar Command Examples 113
Hack 49. Combine gzip, bzip2 with Tar 115
Hack 50. BZ is Eazy! Bz* Command Examples.....116
Hack 51. Cpio Examples 120

Chapter 7: Command Line History 124
Hack 52. Bash Command Line History Examples.....124
Hack 53. History Related Environment Variables.....128
Hack 54. History Expansion Examples 133

Chapter 8: System Administration Tasks.....136
Hack 55. Partition Using fdisk.....136
Hack 56. Format a Partition Using mke2fsk 138
Hack 57. Mount a Partition 140
Hack 58. Fine Tune a Partition Using tune2fs 140
Hack 59. Create a Swap File System.....142
Hack 60. Create a New User.....143
Hack 61. Create a New Group 145
Hack 62. Setup SSH Passwordless Login in OpenSSH 146
Hack 63. Use ssh-copy-id Along With ssh-agent 147
Hack 64. Crontab Examples.....149
Hack 65. Safe Reboot Of Linux Using Magic SysRq Key 151
Hack 66. Linux Parted Command Examples 153
Hack 67. Rsync Command Examples 164
Hack 68. Chkconfig Command Examples.....169
Hack 69. How to Setup Anacron.....174
Hack 70. IPTables Rules Examples 178

Chapter 9: Install Packages.....181
Hack 71. Yum Command Examples.....181
Hack 72. RPM Command Examples.....184
Hack 73. apt-* Command Examples.....188
Hack 74. Install from Source.....190

Chapter 10: LAMP Stack.....192
Hack 75. Install Apache 2 with SSL.....192

Hack 76. Install PHP from Source 196
Hack 77. Install MySQL 199
Hack 78. Install LAMP Stack.....204
Hack 79. Install XAMPP.....210
Hack 80. Secure Your Apache Web Server 212
Hack 81. Apachectl and Httpd Tips.....216
Hack 82. Setup Apache Virtual Host Configuration.....223
Hack 83. Rotate Apache Logs Files.....225

Chapter 11: Bash Scripting.....228
Hack 84. Execution Sequence of .bash_* files.....228
Hack 85. Bash FOR Loops Using C Like Syntax 232
Hack 86. Debug a Shell Script.....234
Hack 87. Quoting.....236
Hack 88. Read Data File Fields Inside a Shell Script 237

Chapter 12: System Monitoring and Performance..239
Hack 89. Free Command.....239
Hack 90. Top Command 240
Hack 91. Df Command 243
Hack 92. Du Command 244
Hack 93. Lsof Commands 245
Hack 94. Vmstat Command 247
Hack 95. Netstat Command 248
Hack 96. Sysctl Command 251
Hack 97. Nice Command 252
Hack 98. Renice Command 254
Hack 99. Kill Command 256
Hack 100. Ps Command.....258
Hack 101. Sar Command 260

Your Support is Appreciated 264
Bash 101 Hacks 264
Sed and Awk 101 Hacks 264
Nagios Core 3 265
Vim 101 Hacks 265

10 Amazing and Essential Linux Books 266

Extended Reading.....269

More Linux Articles.....270

Thank You.....271

Introduction

> *"There are only 10 types of people in the world — those who understand binary, those who don't, and those who understand gray code"*
>
> *— Geek*

There are total of 101 hacks in this book that will help you build a strong foundation in Linux. All the hacks in this book are explained with appropriate Linux command examples that are easy to follow.

This book contains 12 chapters.

- Chapters 1 – 3 explain OpenSSH tips and tricks, CD command hacks, and several essential Linux commands including grep, find and many more.
- Chapters 4 – 6 cover date manipulation, Linux command prompt customization, archive and compression commands. Clear examples are provided.
- Chapter 7 – 9 explain critical Linux sysadmin tasks, package installation on various distros, and bash command line history with clear examples
- Chapter 10 - 12 cover LAMP stack installation and several Linux system monitoring and performance commands with practical examples.

A note on the examples: Most examples are identified in the following way.

Example Description

```
Lines of code for you to type, with the result you will
see on screen.
```

Additional clarification or discussion will appear below the code section in plain text.

About the Author

I'm Ramesh Natarajan, author of The Geek Stuff blog thegeekstuff.com and numerous ebooks including this one.

I have done extensive programming in several languages and C is my favorite. I have done a lot of work on the infrastructure side including Linux system administration, DBA, Networking, Hardware and Storage (EMC).

I also developed passworddragon.com — a free, easy and secure password manager that runs on Windows, Linux and Mac.

Apart from this Linux 101 Hacks eBook, I've also published the following ebooks:

- Vim 101 Hacks
- Nagios Core 3
- Sed and Awk 101 Hacks
- Bash 101 Hacks

If you have any questions while reading this book, don't hesitate to reach out to me. You can connect with me on the following:

- Twitter (@thegeekstuff)
- Facebook page

If you want to write to me directly, use this contact form to reach out to me.

Copyright & Disclaimer

Copyright © 2009 - 2011 – Ramesh Natarajan. All rights reserved. No part of this book may be reproduced, translated, posted or shared in any form, by any means.

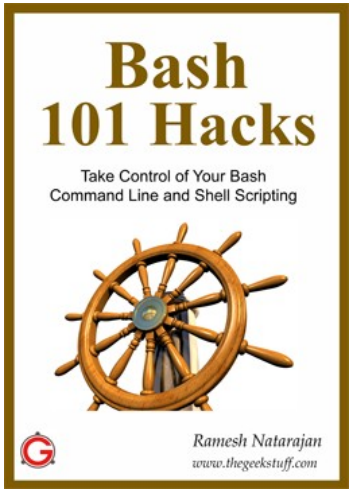
The information provided in this book is provided "as is" with no implied warranties or guarantees.

Version

Version	Date	Revisions
1.0	12 – Feb – 2009	First Edition
2.0	16 – Nov - 2011	Second Edition

More eBooks from The Geek Stuff

Bash 101 Hacks



When you are working on a Linux environment, does any of the following sound familiar to you?

- You are spending significant amount of time doing tasks manually, without knowing how to automate those tasks effectively using scripts.
- You are executing set of commands to complete a task, and retyping the same commands manually with different values to complete similar tasks.
- You are having a good understanding of all Linux commands, but struggling to put them together in a shell script to accomplish a task.

Bash is the default shell on Linux. If you are spending lot of time on Linux environment, you should master the Bash command line features to become efficient.

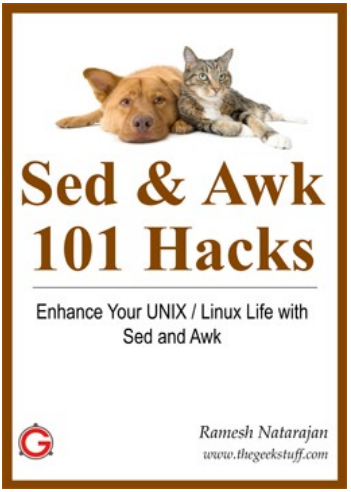
Apart from being an interactive shell, Bash is also a scripting language, which allows you to automate your tasks using Bash shell scripting.

Bash 101 Hacks is a downloadable eBook that contains 101 practical examples on both Bash command line and shell scripting, that will help you understand everything you need to know about Bash.

Get Your Copy of: Bash 101 Hacks eBook

http://www.thegeekstuff.com/bash-101-hacks-ebook/

Sed and Awk 101 Hacks



If you are spending lot of time on UNIX / Linux, the following might sound familiar to you.

- You are manually making the same edits on multiple files. Sometimes the same edits are manually repeated on files on different servers.
- You are constantly viewing large log files (or data files), manually looking for certain lines that contain certain patterns. Once you find those lines, you are manually copying and pasting a few relevant fields from those lines for reporting purposes.
- You are constantly dealing with text files, and spending a lot of time manually manipulating the text files.

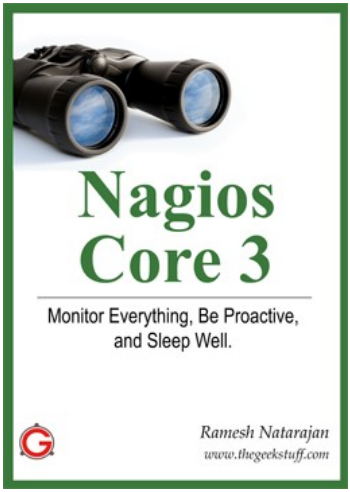
If you are spending lot of time on UNIX / Linux, you'll be manipulating text files frequently. You may be making the similar edits on multiple configuration files on one or more servers. You may be digging huge log files (or data files) looking for certain information.

Sed and Awk 101 Hacks is a downloadable eBook that contains 101 practical examples on various advanced Sed and Awk features, that will help you understand everything you need to know about Sed and Awk.

Get Your Copy of: Sed and Awk 101 Hacks eBook

http://www.thegeekstuff.com/sed-awk-101-hacks-ebook/

Nagios Core



If you are a sysadmin, dba, network administrator, or someone who is responsible for keeping the IT infrastructure up and running, the following might sound familiar to you.

- You don't know when you'll run out of disk space, or when the server will go down, or when the database will crash, or when one of the critical services running on the server will fail.
- You are worried that right people (or team) are not getting notified about the server or services issues at the right time.
- You (or your team) are constantly working on finding and fixing issues as they show up.

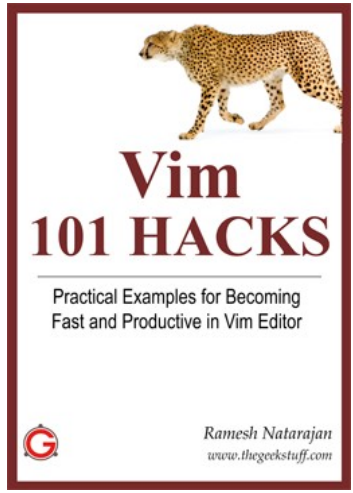
You should implement a robust monitoring solution that will notify you when there is an issue. It should also notify the right people at the right time about a potential issue, even before it becomes critical.

Nagios Core 3 eBook is the only guide you'll ever need to get your IT infrastructure monitored using Nagios Core, and it will help you to understand everything you need to know to implement Nagios Core 3.

Get Your Copy of: Nagios Core eBook

http://www.thegeekstuff.com/nagios-core-ebook/

Vim 101 Hacks



If you are spending lot of time on UNIX / Linux environment, it is essential to become comfortable with the Vim editor.

If you are putting off mastering the Vim editor for a later day because learning Vim editor is not intuitive, friendly, or fun — you are not alone.

Vim editor is very powerful editor that will make you extremely productive once you take some time to learn and understand its features. If you are like most people, you would like to have a structured way of learning this powerful editor and take advantage of all its features.

Vim 101 Hacks is a downloadable eBook that contains 101 practical examples on various advanced Vim features that will make you fast and productive in the Vim editor.

Each hack provided in this eBook is very crisp and easy to understand. The practical examples will show you exactly how to use a particular Vim feature.

Get Your Copy of: Vim 101 Hacks eBook

http://www.thegeekstuff.com/vim-101-hacks-ebook/

Chapter 1: Powerful CD Command Hacks

cd is one of the most frequently used commands during a UNIX session. The 6 cd command hacks mentioned in this chapter will boost your productivity instantly and make it easier to navigate the directory structure from command line.

Hack 1. Define CD Base Directory Using CDPATH

If you are frequently performing cd to subdirectories of a specific parent directory, you can set the CDPATH to the parent directory and perform cd to the subdirectories without giving the parent directory path as explained below.

```
# pwd
/home/ramesh
# cd mail
-bash: cd: mail: No such file or directory

[Note: The above cd is looking for mail directory under
current directory]

# export CDPATH=/etc
# cd mail
/etc/mail

[Note: The above cd is looking for mail under /etc and not
under current directory]

# pwd
/etc/mail
```

To make this change permanent, add export CDPATH=/etc to your ~/.bash_profile

Similar to the PATH variable, you can add more than one directory entry in the CDPATH variable, separating them with : , as shown below.

```
export CDPATH=.:~:/etc:/var
```

This hack can be very helpful under the following situations:

- Oracle DBAs frequently working under $ORACLE_HOME, can set the CDPATH variable to the oracle home
- Unix sysadmins frequently working under /etc, can set the CDPATH variable to /etc
- Developers frequently working under project directory /home/projects, can set the CDPATH variable to /home/projects
- End-users frequently accessing the subdirectories under their home directory, can set the CDPATH variable to ~ (home directory)

Hack 2. Use CD Alias to Navigate Up the Directory

When you are navigating up a very long directory structure, you may be using cd ..\..\ with multiple ..\'s depending on how many directories you want to go up as shown below.

```
# mkdir -p
/tmp/very/long/directory/structure/that/is/too/deep

# cd /tmp/very/long/directory/structure/that/is/too/deep

# pwd
/tmp/very/long/directory/structure/that/is/too/deep

# cd ../../../../
```

```
# pwd
/tmp/very/long/directory/structure
```

Instead of executing cd ../../../.. to navigate four levels up, use one of the following four alias methods:

Method 1: Navigate up the directory using "..n"

In the example below, ..4 is used to go up 4 directory level, ..3 to go up 3 directory level, ..2 to go up 2 directory level. Add the following alias to your ~/.bash_profile and re-login.

```
alias ..="cd .."
alias ..2="cd ../.."
alias ..3="cd ../../.."
alias ..4="cd ../../../.."
alias ..5="cd ../../../../.."
```

```
# cd /tmp/very/long/directory/structure/that/is/too/deep

# ..4
[Note: use ..4 to go up 4 directory level]

# pwd
/tmp/very/long/directory/structure/
```

Method 2: Navigate up the directory using only dots

In the example below, ... (five dots) is used to go up 4 directory level. Typing 5 dots to go up 4 directory structure is really easy to remember, as when you type the first two dots, you are thinking "going up one directory", after that every additional dot, is to go one level up.

So, use .... (four dots) to go up 3 directory level and .. (two dots) to go up 1 directory level. Add the following alias to your ~/.bash_profile and re-login for the ..... (five dots) to work properly.

```
alias ..="cd .."
alias ...="cd ../.."
alias ....="cd ../../.."
alias .....="cd ../../../.."
alias ......="cd ../../../../.."

# cd /tmp/very/long/directory/structure/that/is/too/deep

# .....
[Note: use ..... (five dots) to go up 4 directory level]

# pwd
/tmp/very/long/directory/structure/
```

Method 3: Navigate up the directory using cd followed by consecutive dots

In the example below, cd..... (cd followed by five dots) is used to go up 4 directory level. Making it 5 dots to go up 4 directory structure is really easy to remember, as when you type the first two dots, you are thinking "going up one directory", after that every additional dot, is to go one level up. So, use cd.... (cd followed by four dots) to go up 3 directory level and cd... (cd followed by three dots) to go up 2 directory level. Add the following alias to your ~/.bash_profile and re-login for the above cd..... (five dots) to work properly.

```
alias cd..="cd .."
alias cd...="cd ../.."
alias cd....="cd ../../.."
alias cd.....="cd ../../../.."
alias cd......="cd ../../../../.."
```

```
# cd /tmp/very/long/directory/structure/that/is/too/deep

# cd.....
[Note: use cd..... to go up 4 directory level]

# pwd
/tmp/very/long/directory/structure
```

Method 4: Navigate up the directory using cd followed by number

In the example below, cd4 (cd followed by number 4) is used to go up 4 directory level.

```
alias cd1="cd .."
alias cd2="cd ../.."
alias cd3="cd ../../.."
alias cd4="cd ../../../.."
alias cd5="cd ../../../../.."
```

Hack 3. Perform mkdir and cd Using a Single Command

Sometimes when you create a new directory, you may cd to the new directory immediately to perform some work as shown below.

```
# mkdir -p /tmp/subdir1/subdir2/subdir3

# cd /tmp/subdir1/subdir2/subdir3

# pwd
/tmp/subdir1/subdir2/subdir3
```

Wouldn't it be nice to combine both mkdir and cd in a single command? Add the following to the .bash_profile and re-login.

```
# vi .bash_profile
function mkdircd () { mkdir -p "$@" && eval cd "\"\$$#\""; }
```

Now, perform both mkdir and cd at the same time using a single command as shown below:

```
# mkdircd /tmp/subdir1/subdir2/subdir3

[Note: This creates the directory and cd to it automatically]

# pwd
/tmp/subdir1/subdir2/subdir3
```

Hack 4. Toggle Between Directories

You can toggle between the last two current directories using cd - as shown below.

```
# cd /tmp/very/long/directory/structure/that/is/too/deep
# cd /tmp/subdir1/subdir2/subdir3

# cd -
# pwd
/tmp/very/long/directory/structure/that/is/too/deep



# cd -
# pwd
/tmp/subdir1/subdir2/subdir3
```

```
# cd -
# pwd
/tmp/very/long/directory/structure/that/is/too/deep
```

Hack 5. Manipulate Directory Stack

You can use directory stack to push directories into it and later pop directory from the stack. Following three commands are used in this example.

- dirs: Display the directory stack
- pushd: Push directory into the stack
- popd: Pop directory from the stack and cd to it

Dirs will always print the current directory followed by the content of the stack. Even when the directory stack is empty, dirs command will still print only the current directory as shown below.

```
# popd
-bash: popd: directory stack empty

# dirs
~

# pwd
/home/ramesh
```

How to use pushd and popd? Let us first create some temporary directories and push them to the directory stack as shown below.

```
# mkdir /tmp/dir1
# mkdir /tmp/dir2
# mkdir /tmp/dir3
# mkdir /tmp/dir4
```

```
# cd /tmp/dir1
# pushd .

# cd /tmp/dir2
# pushd .

# cd /tmp/dir3
# pushd .

# cd /tmp/dir4
# pushd .

# dirs
/tmp/dir4 /tmp/dir4 /tmp/dir3 /tmp/dir2 /tmp/dir1

[Note: The first directory (/tmp/dir4) of the dir command
output is always the current directory and not the content
from the stack.]
```

At this stage, the directory stack contains the following directories:

```
/tmp/dir4
/tmp/dir3
/tmp/dir2
/tmp/dir1
```

The last directory that was pushed to the stack will be at the top. When you perform popd, it will cd to the top directory entry in the stack and remove it from the stack. As shown above, the last directory that was pushed into the stack is /tmp/dir4. So, when we do a popd, it will cd to the /tmp/dir4 and remove it from the directory stack as shown below.

```
# popd
```

```
# pwd
/tmp/dir4

[Note: After the above popd, directory Stack Contains:
/tmp/dir3
/tmp/dir2
/tmp/dir1]

# popd
# pwd
/tmp/dir3

[Note: After the above popd, directory Stack Contains:
/tmp/dir2
/tmp/dir1]

# popd
# pwd
/tmp/dir2

[Note: After the above popd, directory Stack Contains:
/tmp/dir1]

# popd
# pwd
/tmp/dir1

[Note: After the above popd, directory Stack is empty!]

# popd
-bash: popd: directory stack empty
```

Hack 6. Automatically Correct Mistyped Directory Names

Use shopt -s cdspell to correct the typos in the cd command automatically as shown below. If you are not good at typing and make lot of mistakes, this will be very helpful.

```
# cd /etc/mall
-bash: cd: /etc/mall: No such file or directory

# shopt -s cdspell

# cd /etc/mall

# pwd
/etc/mail

[Note: By mistake, when I typed mall instead of mail, cd corrected it automatically]
```

Any Questions?

Discuss it here: 6 CD Command Hacks

Chapter 2: Essential Linux Commands

Hack 7. Grep Command

grep command is used to search files for a specific text. This is incredibly powerful command with lots of options.

```
Syntax: grep [options] pattern [files]
```

How can I find all lines matching a specific keyword on a file?

In this example, grep looks for the text John inside /etc/passwd file and displays all the matching lines.

```
# grep John /etc/passwd
jsmith:x:1082:1082:John Smith:/home/jsmith:/bin/bash
jdoe:x:1083:1083:John Doe:/home/jdoe:/bin/bash
```

Option -v, will display all the lines except the match. In the example below, it displays all the records from /etc/password that doesn't match John.

Note: There are several lines in the /etc/password that doesn’t contain the word John. Only the first line of the output is shown below.

```
# grep -v John /etc/passwd
jbourne:x:1084:1084:Jason Bourne:/home/jbourne:/bin/bash
```

How many lines matched the text pattern in a particular file?

In the example below, it displays the total number of lines that contains the text John in /etc/passwd file.

```
# grep -c John /etc/passwd
2
```

You can also get the total number of lines that did not match the specific pattern by passing option -cv.

```
# grep -cv John /etc/passwd
39
```

How to search a text by ignoring the case?

Pass the option -i (ignore case), which will ignore the case while searching.

```
# grep -i john /etc/passwd
jsmith:x:1082:1082:John Smith:/home/jsmith:/bin/bash
jdoe:x:1083:1083:John Doe:/home/jdoe:/bin/bash
```

How do I search all subdirectories for a text matching a specific pattern?

Use option -r (recursive) for this purpose. In the example below, it will search for the text "John" by ignoring the case inside all the subdirectories under /home/users.

This will display the output in the format of "filename: line that matching the pattern". You can also pass the option -l, which will display only the name of the file that matches the pattern.

```
# grep -ri john /home/users
/home/users/subdir1/letter.txt:John, Thanks for your
contribution.
/home/users/name_list.txt:John Smith
/home/users/name_list.txt:John Doe


# grep -ril john /root
```

```
/home/users/subdir1/letter.txt
/home/users/name_list.txt
```

Additional Grep Examples:

Get a Grip on the Grep! – 15 Practical Grep Command Examples

The Power of Z Commands – Zcat, Zless, Zgrep, Zdiff Examples

7 Linux Grep OR, Grep AND, Grep NOT Operator Examples

Hack 8. Regular Expression in Grep

Regular expressions are used to search and manipulate the text, based on the patterns. Most of the Linux commands and programming languages use regular expression.

This hack explains how to use most frequently used reg-ex operators in Grep command.

Beginning of line (^)

In grep command, caret Symbol ^ matches the expression at the start of a line. In the following example, it displays all the line which starts with the Nov 10. i.e All the messages logged on November 10.

```
$ grep "^Nov 10" messages.1
Nov 10 01:12:55 gs123 ntpd[2241]: time reset +0.177479 s
Nov 10 01:17:17 gs123 ntpd[2241]: synchronized to
LOCAL(0), stratum 10
Nov 10 01:18:49 gs123 ntpd[2241]: synchronized to
15.1.13.13, stratum 3
Nov 10 13:21:26 gs123 ntpd[2241]: time reset +0.146664 s
```

```
Nov 10 13:25:46 gs123 ntpd[2241]: synchronized to
LOCAL(0), stratum 10
Nov 10 13:26:27 gs123 ntpd[2241]: synchronized to
15.1.13.13, stratum 3
```

The ^ matches the expression in the beginning of a line, only if it is the first character in a regular expression. ^N matches line beginning with N.

End of the line ($)

Character $ matches the expression at the end of a line. The following command will help you to get all the lines which ends with the word “terminating”.

```
$ grep "terminating.$" messages
Jul 12 17:01:09 cloneme kernel: Kernel log daemon
terminating.
Oct 28 06:29:54 cloneme kernel: Kernel log daemon
terminating.
```

From the above output you can come to know when all the kernel log has got terminated. Just like ^ matches the beginning of the line only if it is the first character, $ matches the end of the line only if it is the last character in a regular expression.

Count of empty lines (^$)

Using ^ and $ character you can find out the empty lines available in a file. “^$” specifies empty line.

```
$ grep -c  "^$" messages anaconda.log
messages:0
anaconda.log:3
```

The above commands displays the count of the empty lines available in the messages and anaconda.log files.

Single Character (.)

The special meta-character “.” (dot) matches any character except the end of the line character. Let us take the input file which has the content as follows.

```
$ cat input
1. first line
2. hi hello
3. hi zello how are you
4. cello
5. aello
6. eello
7. last line
```

Now let us search for a word which has any single character followed by ello. i.e hello, cello etc.,

```
$ grep ".ello" input
2. hi hello
3. hi zello how are you
4. cello
5. aello
6. eello
```

In case if you want to search for a word which has only 4 character you can give grep -w “….” where single dot represents any single character.

Zero or more occurrence (*)

The special character “*” matches zero or more occurrence of the previous character. For example, the pattern ’1*’ matches zero or more ’1′.

The following example searches for a pattern “kernel: *” i.e kernel: and zero or more occurrence of space character.

```
 $ grep "kernel: *." *
messages.4:Jul 12 17:01:02 cloneme kernel: ACPI: PCI
interrupt for device 0000:00:11.0 disabled
messages.4:Oct 28 06:29:49 cloneme kernel: ACPI: PM-Timer
IO Port: 0x1008
messages.4:Oct 28 06:31:06 btovm871 kernel:  sda: sda1
sda2 sda3
messages.4:Oct 28 06:31:06 btovm871 kernel: sd 0:0:0:0:
Attached scsi disk sda
.
.
```

In the above example it matches for kernel and colon symbol followed by any number of spaces/no space and “.” matches any single character.

Additional Grep Reg-Ex Examples:

Regular Expressions in Grep Command with 10 Examples – Part I

Advanced Regular Expressions in Grep Command with 10 Examples – Part II

Hack 9. Find Command

find is frequently used command to find files in the UNIX filesystem based on numerous conditions. Let us review some practice examples of find command.

```
Syntax: find [pathnames] [conditions]
```

How to find files containing a specific word in its name?

The following command looks for all the files under /etc directory with mail in the filename.

```
# find /etc -name "*mail*"
```

How to find all the files greater than certain size?

The following command will list all the files in the system greater than 100MB.

```
# find / -type f -size +100M
```

How to find files that are not modified in the last x number of days?

The following command will list all the files that were modified more than 60 days ago under the current directory.

```
# find . -mtime +60
```

How to find files that are modified in the last x number of days?

The following command will list all the files that were modified in the last two days under the current directory.

```
# find . –mtime -2
```

How to delete all the archive files with extension *.tar.gz and greater than 100MB?

Please be careful while executing the following command as you don't want to delete the files by mistake. The best practice is to execute the same command with ls –l to make sure you know which files will get deleted when you execute the command with rm.

```
# find / -type f -name *.tar.gz -size +100M -exec ls -l {}
\;


# find / -type f -name *.tar.gz -size +100M -exec rm -f {}
\;
```

How to archive all the files that are not modified in the last x number of days?

The following command finds all the files not modified in the last 60 days under /home/jsmith directory and creates an archive files under /tmp in the format of ddmmyyyy_archive.tar.

```
# find /home/jsmith -type f -mtime +60 | xargs tar -cvf
/tmp/`date '+%d%m%Y'_archive.tar`
```

Additional Find Examples:

Mommy, I found it! – 15 Practical Linux Find Command Examples

Daddy, I found it!, 15 Awesome Linux Find Command Examples (Part2)

Hack 10. Suppress Standard Output and Error Message

Sometime while debugging a shell script, you may not want to see either the standard output or standard error message. Use /dev/null as shown below for suppressing the output.

Suppress standard output using > /dev/null

This will be very helpful when you are debugging shell scripts, where you don't want to display the echo statement and interested in only looking at the error messages.

```
# cat file.txt > /dev/null

# ./shell-script.sh > /dev/null
```

Suppress standard error using 2> /dev/null

This is also helpful when you are interested in viewing only the standard output and don't want to view the error messages.

```
# cat invalid-file-name.txt 2> /dev/null


# ./shell-script.sh 2> /dev/null
```

Note: One of the most effective ways to use this is in the crontab, where you can suppress the output and error message of a cron task as shown below.

```
30 1 * * * command > /dev/null 2>&1
```

Hack 11. Join Command

Join command combines lines from two files based on a common field.

In the example below, we have two files – employee.txt and salary.txt. Both have employee-id as common field. So, we can use join command to combine the data from these two files using employee-id as shown below.

```
$ cat employee.txt
100 Jason Smith
200 John Doe
300 Sanjay Gupta
400 Ashok Sharma

$ cat bonus.txt
100 $5,000
200 $500
300 $3,000
400 $1,250
```

```
$ join employee.txt bonus.txt
100 Jason Smith $5,000
200 John Doe $500
300 Sanjay Gupta $3,000
400 Ashok Sharma $1,250
```

Hack 12. Change the Case

Convert a file to all upper-case

```
$ cat employee.txt
100 Jason Smith
200 John Doe
300 Sanjay Gupta
400 Ashok Sharma

$ tr a-z A-Z < employee.txt
100 JASON SMITH
200 JOHN DOE
300 SANJAY GUPTA
400 ASHOK SHARMA
```

Convert a file to all lower-case

```
$ cat department.txt
100 FINANCE
200 MARKETING
300 PRODUCT DEVELOPMENT
400 SALES
```

```
$ tr A-Z a-z < department.txt
100 finance
200 marketing
300 product development
400 sales
```

Hack 13. Xargs Command

xargs is a very powerful command that takes output of a command and pass it as argument of another command.

The following are some practical examples on how to use xargs effectively.

1. When you are trying to delete too many files using rm, you may get error message: /bin/rm Argument list too long – Linux. Use xargs to avoid this problem.

```
find ~ -name '*.log' -print0 | xargs -0 rm -f
```

2. Get a list of all the *.conf file under /etc/. There are different ways to get the same result. Following example is only to demonstrate the use of xargs. The output of the find command in this example is passed to the ls –l one by one using xargs.

```
# find /etc -name "*.conf" | xargs ls –l
```

3. If you have a file with list of URLs that you would like to download, you can use xargs as shown below.

```
# cat url-list.txt | xargs wget –c
```

4. Find out all the jpg images and archive it.

```
# find / -name *.jpg -type f -print | xargs tar -cvzf
images.tar.gz
```

5. Copy all the images to an external hard-drive.

```
# ls *.jpg | xargs -n1 -i cp {} /external-hard-
drive/directory
```

Hack 14. Sort Command

Sort command sorts the lines of a text file. Following are several practical examples on how to use the sort command based on the following sample text file that has employee information in the format:

```
employee_name:employee_id:department_name.
```

```
$ cat names.txt
Emma Thomas:100:Marketing
Alex Jason:200:Sales
Madison Randy:300:Product Development
Sanjay Gupta:400:Support
Nisha Singh:500:Sales
```

Sort a text file in ascending order

```
$ sort names.txt
Alex Jason:200:Sales
Emma Thomas:100:Marketing
Madison Randy:300:Product Development
Nisha Singh:500:Sales
Sanjay Gupta:400:Support
```

Sort a text file in descending order

```
$ sort -r names.txt
Sanjay Gupta:400:Support
Nisha Singh:500:Sales
Madison Randy:300:Product Development
Emma Thomas:100:Marketing
Alex Jason:200:Sales
```

Sort a colon delimited text file on 2nd field (employee_id)

```
$ sort -t: -k 2 names.txt
Emma Thomas:100:Marketing
Alex Jason:200:Sales
Madison Randy:300:Product Development
Sanjay Gupta:400:Support
Nisha Singh:500:Sales
```

Sort a tab delimited text file on 3rd field (department_name) and suppress duplicates

```
$ sort -t: -u -k 3 names.txt
Emma Thomas:100:Marketing
Madison Randy:300:Product Development
Alex Jason:200:Sales
Sanjay Gupta:400:Support
```

Sort the passwd file by the 3rd field (numeric userid)

```
$ sort -t: -k 3n /etc/passwd | more
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
adm:x:3:4:adm:/var/adm:/sbin/nologin
```

```
lp:x:4:7:lp:/var/spool/lpd:/sbin/nologin
```

Sort /etc/hosts file by ip-address

```
$ sort -t . -k 1,1n -k 2,2n -k 3,3n -k 4,4n /etc/hosts
127.0.0.1 localhost.localdomain localhost
192.168.100.101 dev-db.thegeekstuff.com dev-db
192.168.100.102 prod-db.thegeekstuff.com prod-db
192.168.101.20  dev-web.thegeekstuff.com dev-web
192.168.101.21  prod-web.thegeekstuff.com prod-web
```

Combine sort with other commands

- **ps –ef | sort** : Sort the output of process list
- **ls -al | sort +4n** : List the files in the ascending order of the file-size. i.e sorted by 5th filed and displaying smallest files first.
- **ls -al | sort +4nr** : List the files in the descending order of the file-size. i.e sorted by 5th filed and displaying largest files first.

Hack 15. Uniq Command

Uniq command is mostly used in combination with sort command, as uniq removes duplicates only from a sorted file. i.e In order for uniq to work, all the duplicate entries should be in the adjacent lines. The following are some common examples.

1. When you have an employee file with duplicate entries, you can do the following to remove duplicates.

```
$ sort namesd.txt | uniq


$ sort –u namesd.txt
```

2. If you want to know how many lines are duplicates, do the following. The first field in the following examples indicates how many duplicates where found for that particular line. So, in this example the lines beginning with Alex and Emma were found twice in the namesd.txt file.

```
$ sort namesd.txt | uniq –c
      2 Alex Jason:200:Sales
      2 Emma Thomas:100:Marketing
      1 Madison Randy:300:Product Development
      1 Nisha Singh:500:Sales
      1 Sanjay Gupta:400:Support
```

3. The following displays only the entries that are duplicates.

```
$ sort namesd.txt | uniq –cd
      2 Alex Jason:200:Sales
      2 Emma Thomas:100:Marketing
```

Hack 16. Cut Command

Cut command can be used to display only specific columns from a text file or other command outputs.

The following are some of the examples.

Display the 1st field (employee name) from a colon delimited file

```
$ cut -d: -f 1 names.txt
Emma Thomas
Alex Jason
Madison Randy
Sanjay Gupta
Nisha Singh
```

Display 1st and 3rd field from a colon delimited file

```
$ cut -d: -f 1,3 names.txt
Emma Thomas:Marketing
Alex Jason:Sales
Madison Randy:Product Development
Sanjay Gupta:Support
Nisha Singh:Sales
```

Display only the first 8 characters of every line in a file

```
$ cut -c 1-8 names.txt
Emma Tho
Alex Jas
Madison
Sanjay G
Nisha Si
```

Misc Cut command examples

- **cut -d: -f1 /etc/passwd** Displays the unix login names for all the users in the system.
- **free | tr -s ' ' | sed '/^Mem/!d' | cut -d" " -f2** Displays the total memory available on the system.

Hack 17. Stat Command

Stat command can be used either to check the status/properties of a single file or the filesystem.

Display statistics of a file or directory.

```
$ stat /etc/my.cnf
 File: `/etc/my.cnf'
```

```
  Size: 346 Blocks: 16 IO Block: 4096   regular file
Device: 801h/2049d      Inode: 279856      Links: 1
Access: (0644/-rw-r--r--)  Uid: (0/root)   Gid: (0/root)
Access: 2009-01-01 02:58:30.000000000 -0800
Modify: 2006-06-01 20:42:27.000000000 -0700
Change: 2007-02-02 14:17:27.000000000 -0800


$ stat /home/ramesh
 File: `/home/ramesh'
  Size: 4096            Blocks: 8          IO Block: 4096
directory
Device: 803h/2051d      Inode: 5521409     Links: 7
Access: (0755/drwxr-xr-x)  Uid: (401/ramesh)   Gid:
(401/ramesh)
Access: 2009-01-01 12:17:42.000000000 -0800
Modify: 2009-01-01 12:07:33.000000000 -0800
Change: 2009-01-09 12:07:33.000000000 -0800
```

Display the status of the filesystem using option –f

```
$ stat -f /
 File: "/"
    ID: 0        Namelen: 255     Type: ext2/ext3
Blocks: Total: 2579457    Free: 2008027    Available:
1876998    Size: 4096
Inodes: Total: 1310720    Free: 1215892
```

Additional Stat Examples:

Unix Stat Command: How To Identify File Attributes

Hack 18. Diff Command

diff command compares two different files and reports the difference. The output of the diff command is very cryptic and not straight forward to read.

```
Syntax: diff [options] file1 file2
```

What was modified in my new file when compare to my old file?

The option -w in the diff command will ignore the white space while performing the comparison.

In the following diff output:

- The lines above ---, indicates the changes happened in first file in the diff command (i.e name_list.txt).
- The lines below ---, indicates the changes happened to the second file in the diff command (i.e name_list_new.txt). The lines that belong to the first file starts with < and the lines of second file starts with >.

```
# diff -w name_list.txt name_list_new.txt
2c2,3
< John Doe
---
> John M Doe
> Jason Bourne
```

Additional Diff Examples:

Top 4 File Difference Tools on Linux – Diff, Colordiff, Wdiff, Vimdiff

Visual File Diff with Vimdiff – It Does Make a Difference!

Hack 19. Display Total Connect Time of Users

Ac command will display the statistics about the user's connect time.

Connect time for the current logged in user

With the option –d, it will break down the output for the individual days. In this example, I've been logged in to the system for more than 6 hours today. On Dec 1st, I was logged in for about 1 hour.

```
$ ac –d
Dec  1  total        1.08
Dec  2  total        0.99
Dec  3  total        3.39
Dec  4  total        4.50
Today   total        6.10
```

Connect time for all the users

To display connect time for all the users use –p as shown below. Please note that this indicates the cumulative connect time for the individual users.

```
$ ac -p
        john                                    3.64
        madison                                 0.06
        sanjay                                  88.17
        nisha                                   105.92
        ramesh                                  111.42
        total 309.21
```

Connect time for a specific user

To get a connect time report for a specific user, execute the following:

```
$ ac -d sanjay
Jul  2  total       12.85
Aug 25  total        5.05
Sep  3  total        1.03
Sep  4  total        5.37
Dec 24  total        8.15
Dec 29  total        1.42
Today   total        2.95
```

Hack 20. Execute Commands in the Background

You can use one of the 5 methods explained in this hack to execute a Linux command, or shell script in the background.

Method 1. Use &

You can execute a command (or shell script) as a background job by appending an ampersand to the command as shown below.

```
$ ./my-shell-script.sh &
```

Method 2. Nohup

After you execute a command (or shell script) in the background using &, if you logout from the session, the command will get killed. To avoid that, you should use nohup as shown below.

```
$ nohup ./my-shell-script.sh &
```

Method 3. Screen Command

After you execute a command in the background using nohup and &, the command will get executed even after you logout. But, you cannot

connect to the same session again to see exactly what is happening on the screen. To do that, you should use screen command.

Linux screen command offers the ability to detach a session that is running some process, and then attach it at a later time. When you reattach the session later, your terminals will be there exactly in the way you left them earlier.

Method 4. At Command

Using at command you can schedule a job to run at a particular date and time. For example, to execute the backup script at 10 a.m tomorrow, do the following.

```
$ at -f backup.sh 10 am tomorrow
```

Method 5. Watch Command

To execute a command continuously at a certain interval, use watch command as shown below.

```
$ watch df -h
```

Additional Background Command Examples:

Bg, Fg, &, Ctrl-Z – 5 Examples to Manage Unix Background Jobs

Unix Nohup: Run a Command or Shell-Script Even after You Logout

Screen Command Examples: Get Control of Linux / Unix Terminal

at, atq, atrm, batch Commands using 9 Examples

Repeat Unix Commands or Shell-Scripts every N seconds

Hack 21. Sed Basics - Find and Replace Using RegEx

This hack explains how to use sed substitute command “s”.

The `s’ command is probably the most important in `sed’ and has a lot of different options.

The `s’ command attempts to match the pattern space against the supplied REGEXP; if the match is successful, then that portion of the pattern space which was matched is replaced with REPLACEMENT.

Syntax:

```
#sed 'ADDRESSs/REGEXP/REPLACEMENT/FLAGS' filename
#sed 'PATTERNs/REGEXP/REPLACEMENT/FLAGS' filename
```

- s is substitute command
- / is a delimiter
- REGEXP is regular expression to match
- REPLACEMENT is a value to replace

FLAGS can be any of the following :

- g Replace all the instance of REGEXP with REPLACEMENT
- n Could be any number,replace nth instance of the REGEXP with REPLACEMENT.
- p If substitution was made, then prints the new pattern space.
- i match REGEXP in a case-insensitive manner.
- w file If substitution was made, write out the result to the given file.

- We can use different delimiters (one of @ % ; :) instead of /

Let us first create thegeekstuff.txt file that will be used in all the examples mentioned below.

```
$ cat thegeekstuff.txt
# Instruction Guides
1. Linux Sysadmin, Linux Scripting etc.
2. Databases - Oracle, mySQL etc.
3. Security (Firewall, Network, Online Security etc)
4. Storage in Linux
5. Productivity (Too many technologies to explore, not
much time available)
#  Additional FAQS
6. Windows- Sysadmin, reboot etc.
```

Substitute Word “Linux” to “Linux-Unix” Using sed s//

In the example below, in the output line “1. Linux-Unix Sysadmin, Linux Scripting etc” only first Linux is replaced by Linux-Unix. If no flags are specified the first match of line is replaced.

```
$ sed 's/Linux/Linux-Unix/' thegeekstuff.txt
# Instruction Guides
1. Linux-Unix Sysadmin, Linux Scripting etc.
2. Databases - Oracle, mySQL etc.
3. Security (Firewall, Network, Online Security etc)
4. Storage in Linux-Unix
5. Productivity (Too many technologies to explore, not
much time available)
#  Additional FAQS
6. Windows- Sysadmin, reboot etc.
```

Substitute all Appearances of a Word Using sed s//g

The below sed command replaces all occurrences of Linux to Linux-Unix using global substitution flag “g”.

```
$ sed 's/Linux/Linux-Unix/g' thegeekstuff.txt
# Instruction Guides
1. Linux-Unix Sysadmin, Linux-Unix Scripting etc.
2. Databases - Oracle, mySQL etc.
3. Security (Firewall, Network, Online Security etc)
4. Storage in Linux-Unix
5. Productivity (Too many technologies to explore, not
much time available)
#  Additional FAQS
6. Windows- Sysadmin, reboot etc.
```

Substitute Only 2nd Occurrence of a Word Using sed s//2

In the example below, in the output line “1. Linux Sysadmin, Linux-Unix Scripting etc.” only 2nd occurrence of Linux is replaced by Linux-Unix.

```
$ sed 's/Linux/Linux-Unix/2' thegeekstuff.txt
# Instruction Guides
1. Linux Sysadmin, Linux-Unix Scripting etc.
2. Databases - Oracle, mySQL etc.
3. Security (Firewall, Network, Online Security etc)
4. Storage in Linux
5. Productivity (Too many technologies to explore, not
much time available)
#  Additional FAQS
6. Windows- Sysadmin, reboot etc.
4. Write Changes to a File and Print the Changes Using sed
s//gpw
```

The example below has substitution with three flags. It substitutes all the occurrence of Linux to Linux-Unix and prints the substituted output as well as written the same to the given the file.

```
$ sed -n 's/Linux/Linux-Unix/gpw output' thegeekstuff.txt
1. Linux-Unix Sysadmin, Linux-Unix Scripting etc.
4. Storage in Linux-Unix


$ cat output
1. Linux-Unix Sysadmin, Linux-Unix Scripting etc.
4. Storage in Linux-Unix
5. Substitute Only When the Line Matches with the Pattern
Using sed
```

In this example, if the line matches with the pattern “-”, then it replaces all the characters from “-” with the empty.

```
$ sed '/\-/s/\-.*//g' thegeekstuff.txt
# Instruction Guides
1. Linux Sysadmin, Linux Scripting etc.
2. Databases
3. Security (Firewall, Network, Online Security etc)
4. Storage in Linux
5. Productivity (Too many technologies to explore, not
much time available)
#  Additional FAQS
6. Windows
```

Delete Last X Number of Characters From Each Line Using sed

This sed example deletes last 3 characters from each line.

```
$ sed 's/...$//' thegeekstuff.txt
# Instruction Gui
1. Linux Sysadmin, Linux Scripting e
2. Databases - Oracle, mySQL e
3. Security (Firewall, Network, Online Security e
```

```
4. Storage in Li
5. Productivity (Too many technologies to explore, not
much time availab
#  Additional F
6. Windows- Sysadmin, reboot e
```

Eliminate Comments Using sed

Delete all the comment lines from a file as shown below using sed command.

```
$  sed -e 's/#.*//' thegeekstuff.txt


1. Linux Sysadmin, Linux Scripting etc.
2. Databases - Oracle, mySQL etc.
3. Security (Firewall, Network, Online Security etc)
4. Storage in Linux
5. Productivity (Too many technologies to explore, not
much time available)


6. Windows- Sysadmin, reboot etc.
8. Eliminate Comments and Empty Lines Using sed
```

In the following example, there are two commands separated by ';'

First command replaces the lines starting with the # to the blank lines

Second command deletes the empty lines.

```
$ sed -e 's/#.*//;/^$/d'  thegeekstuff.txt
1. Linux Sysadmin, Linux Scripting etc.
2. Databases - Oracle, mySQL etc.
3. Security (Firewall, Network, Online Security etc)
4. Storage in Linux
```

```
5. Productivity (Too many technologies to explore, not
much time available)
6. Windows- Sysadmin, reboot etc.
9. Convert DOS newlines (CR/LF) to Unix format Using sed
```

Eliminate HTML Tags from file Using sed

In this example, the regular expression given in the sed command matches the html tags and replaces with the empty.

```
$ sed -e 's/<[^>]*>//g'
This <b> is </b> an <i>example</i>.
This  is  an example.
```

Any Questions?

Discuss it here: Sed Tutorial: Find and Replace Text Inside a File Using RegEx

Additional Sed Substitution Examples:

Advanced Sed Substitution Examples

Hack 22: Awk Introduction – Print Examples

This hack explains the fundamental awk working methodology along with 7 practical awk print examples.

Awk Introduction and Printing Operations

Awk is a programming language which allows easy manipulation of structured data and the generation of formatted reports. Awk stands for the names of its authors “Aho, Weinberger, and Kernighan”

The Awk is mostly used for pattern scanning and processing. It searches one or more files to see if they contain lines that matches with the specified patterns and then perform associated actions.

Some of the key features of Awk are:

- Awk views a text file as records and fields.
- Like common programming language, Awk has variables, conditionals and loops
- Awk has arithmetic and string operators.
- Awk can generate formatted reports
- Awk reads from a file or from its standard input, and outputs to its standard output. Awk does not get along with non-text files.

Syntax:

```
awk '/search pattern1/ {Actions}
     /search pattern2/ {Actions}' file
```

In the above awk syntax:

- search pattern is a regular expression.
- Actions – statement(s) to be performed.
- several patterns and actions are possible in Awk.
- file – Input file.
- Single quotes around program is to avoid shell not to interpret any of its special characters.

Awk Working Methodology

1. Awk reads the input files one line at a time.
2. For each line, it matches with given pattern in the given order, if matches performs the corresponding action.
3. If no pattern matches, no action will be performed.
4. In the above syntax, either search pattern or action are optional, But not both.

5. If the search pattern is not given, then Awk performs the given actions for each line of the input.
6. If the action is not given, print all that lines that matches with the given patterns which is the default action.
7. Empty braces with out any action does nothing. It wont perform default printing operation.
8. Each statement in Actions should be delimited by semicolon.

Let us create employee.txt file which has the following content, which will be used in the examples mentioned below.

```
$ cat employee.txt
100  Thomas  Manager    Sales       $5,000
200  Jason   Developer  Technology  $5,500
300  Sanjay  Sysadmin   Technology  $7,000
400  Nisha   Manager    Marketing   $9,500
500  Randy   DBA        Technology  $6,000
```

Default behavior of Awk

By default Awk prints every line from the file.

```
$ awk '{print;}' employee.txt
100  Thomas  Manager    Sales       $5,000
200  Jason   Developer  Technology  $5,500
300  Sanjay  Sysadmin   Technology  $7,000
400  Nisha   Manager    Marketing   $9,500
500  Randy   DBA        Technology  $6,000
```

In the above example pattern is not given. So the actions are applicable to all the lines.

Action print with out any argument prints the whole line by default. So it prints all the lines of the file with out fail. Actions has to be enclosed with in the braces.

Print the lines which matches with the pattern.

```
$ awk '/Thomas/
> /Nisha/' employee.txt
100  Thomas  Manager    Sales       $5,000
400  Nisha   Manager    Marketing   $9,500
```

In the above example it prints all the line which matches with the 'Thomas' or 'Nisha'. It has two patterns. Awk accepts any number of patterns, but each set (patterns and its corresponding actions) has to be separated by newline.

Print only specific field

Awk has number of built in variables. For each record i.e line, it splits the record delimited by whitespace character by default and stores it in the $n variables. If the line has 4 words, it will be stored in $1, $2, $3 and $4. $0 represents whole line. NF is a built in variable which represents total number of fields in a record.

```
$ awk '{print $2,$5;}' employee.txt
Thomas $5,000
Jason $5,500
Sanjay $7,000
Nisha $9,500
Randy $6,000

$ awk '{print $2,$NF;}' employee.txt
Thomas $5,000
Jason $5,500
Sanjay $7,000
Nisha $9,500
Randy $6,000
```

In the above example $2 and $5 represents Name and Salary respectively. We can get the Salary using $NF also, where $NF represents last field. In the print statement ',' is a concatenator.

Initialization and Final Action

Awk has two important patterns which are specified by the keyword called BEGIN and END.

Syntax:

```
BEGIN { Actions}
{ACTION} # Action for everyline in a file
END { Actions }
```

- # is for comments in Awk
- Actions specified in the BEGIN section will be executed before starts reading the lines from the input.
- END actions will be performed after completing the reading and processing the lines from the input.

```
$ awk 'BEGIN {print
"Name\tDesignation\tDepartment\tSalary";}
> {print $2,"\t",$3,"\t",$4,"\t",$NF;}
> END{print "Report Generated\n--------------";
> }' employee.txt
Name        Designation Department  Salary
Thomas      Manager     Sales       $5,000
Jason       Developer   Technology  $5,500
Sanjay      Sysadmin    Technology  $7,000
Nisha       Manager     Marketing   $9,500
Randy       DBA         Technology  $6,000
Report Generated
--------------
```

In the above example, it prints headline and last file for the reports.

Find the employees who has employee id greater than 200

```
$ awk '$1 >200' employee.txt
300  Sanjay  Sysadmin   Technology  $7,000
400  Nisha   Manager    Marketing   $9,500
500  Randy   DBA        Technology  $6,000
```

In the above example, first field ($1) is employee id. So if $1 is greater than 200, then just do the default print action to print the whole line.

Print the list of employees in Technology department

Now department name is available as a fourth field, so need to check if $4 matches with the string “Technology”, if yes print the line.

```
$ awk '$4 ~/Technology/' employee.txt
200  Jason   Developer  Technology  $5,500
300  Sanjay  Sysadmin   Technology  $7,000
500  Randy   DBA        Technology  $6,000
```

Operator ~ is for comparing with the regular expressions. If it matches the default action i.e print whole line will be performed.

Print number of employees in Technology department

The below example, checks if the department is Technology, if it is yes, in the Action, just increment the count variable, which was initialized with zero in the BEGIN section.

```
$ awk 'BEGIN { count=0;}
$4 ~ /Technology/ { count++; }
END { print "Number of employees in Technology Dept
=",count;}' employee.txt
Number of employees in Tehcnology Dept = 3
```

Then at the end of the process, just print the value of count which gives you the number of employees in Technology department.

Any Questions?

Discuss it here: Awk Introduction Tutorial – 7 Awk Print Examples

Additional Awk Examples:

Understand Awk Variables with 3 Practical Examples

8 Powerful Awk Built-in Variables – FS, OFS, RS, ORS, NR, NF, FILENAME, FNR

7 Powerful Awk Operators Examples (Unary, Binary, Arithmetic, String, Assignment, Conditional, Reg-Ex Awk Operators)

AWK Arrays Explained with 5 Practical Examples

Hack 23. Vim Editor Navigation Fundamentals

Navigation is a vital part of text editing. To be very productive, you should be aware of all possible navigation shortcuts in your editor.

This hack explains the following 8 Vi / Vim navigation options.

1. Line navigation
2. Screen navigation
3. Word navigation
4. Special navigation
5. Paragraph navigation
6. Search navigation

7. Code navigation
8. Navigation from command line

Line Navigation

Following are the four navigation that can be done line by line.

- k – navigate upwards
- j – navigate downwards
- l – navigate right side
- h – navigate left side

By using the repeat factor in VIM we can do this operation for N times. For example, when you want to go down by 10 lines, then type “10j”.

Within a line if you want to navigate to different position, you have 4 other options.

- 0 – go to the starting of the current line.
- ^ – go to the first non blank character of the line.
- $ – go to the end of the current line.
- g_ – go to the last non blank character of the line.

Screen Navigation

Following are the three navigation which can be done in relation to text shown in the screen.

- H – Go to the first line of current screen.
- M – Go to the middle line of current screen.
- L – Go to the last line of current screen.
- ctrl+f – Jump forward one full screen.
- ctrl+b – Jump backwards one full screen
- ctrl+d – Jump forward (down) a half screen
- ctrl+u – Jump back (up) one half screen

Special Navigation

You may want to do some special navigation inside a file, which are:

- N% – Go to the Nth percentage line of the file.
- NG – Go to the Nth line of the file.
- G – Go to the end of the file.
- `” – Go to the position where you were in NORMAL MODE while last closing the file.
- `^ – Go to the position where you were in INSERT MODE while last closing the file.
- g – Go to the beginning of the file.

Word Navigation

You may want to do several navigation in relation to the words, such as:

- e – go to the end of the current word.
- E – go to the end of the current WORD.
- b – go to the previous (before) word.
- B – go to the previous (before) WORD.
- w – go to the next word.
- W – go to the next WORD.

WORD – WORD consists of a sequence of non-blank characters, separated with white space.

word – word consists of a sequence of letters, digits and underscores.

Example to show the difference between WORD and word

- 192.168.1.1 – single WORD
- 192.168.1.1 – seven words.

Paragraph Navigation

- { – Go to the beginning of the current paragraph. By pressing { again and again move to the previous paragraph beginnings.
- } – Go to the end of the current paragraph. By pressing } again and again move to the next paragraph end, and again.

Search Navigation

- /i – Search for a pattern which will you take you to the next occurrence of it.
- ?i – Search for a pattern which will you take you to the previous occurrence of it.
- * - Go to the next occurrence of the current word under the cursor.
- # - Go to the previous occurrence of the current word under the cursor.

Code Navigation

- % – Go to the matching braces, or parenthesis inside code.

Navigation from Command Line

Vim +N filename: Go to the Nth line of the file after opening it.

```
vim +10 /etc/passwd
```

Vim +/pattern filename: Go to the particular pattern's line inside the file, first occurrence from first. In the following example, it will open the README file and jump to the first occurrence of the word "install".

```
vim +/install README
```

Vim +?patten filename: Go to the particular pattern's line inside the file, first occurrence from last. In the following example, it will open the README file and jump to the last occurrence of the word "bug".

```
vim +?bug README
```

Any Questions?

Discuss it here: 8 Essential Vim Editor Navigation Fundamentals

More Vim Examples:

Vim search and replace – 12 powerful find and replace examples.

How To add bookmarks inside the Vim editor

How To record and play inside the Vim editor

Correct spelling mistakes automatically inside the Vim Editor

Hack 24. Chmod Command Examples

This hack explains how to use symbolic representation with chmod.

Following are the symbolic representation of three different roles:

- u is for user,
- g is for group,
- and o is for others.

Following are the symbolic representation of three different permissions:

- r is for read permission,
- w is for write permission,
- x is for execute permission.

The following are few examples on how to use the symbolic representation on chmod.

Add single permission to a file/directory

Changing permission to a single set. + symbol means adding permission. For example, do the following to give execute permission for the user irrespective of anything else:

```
$ chmod u+x filename
```

Add multiple permission to a file/directory

Use comma to separate the multiple permission sets as shown below.

```
$ chmod u+r,g+x filename
```

Remove permission from a file/directory

The following example removes read and write permission for the user.

```
$ chmod u-rx filename
```

Change permission for all roles on a file/directory

The following example assigns execute privilege to user, group and others (basically anybody can execute this file).

```
$ chmod a+x filename
```

Make permission for a file same as another file (using reference)

If you want to change a file permission same as another file, use the reference option as shown below. In this example, file2′s permission will be set exactly same as file1′s permission.

```
$ chmod --reference=file1 file2
```

Apply the permission to all the files under a directory recursively

Use option -R to change the permission recursively as shown below.

```
$ chmod -R 755 directory-name/
```

Change execute permission only on the directories (files are not affected)

On a particular directory if you have multiple sub-directories and files, the following command will assign execute permission only to all the sub-directories in the current directory (not the files in the current directory).

```
$ chmod u+X *
```

Note: If the files has execute permission already for either the group or others, the above command will assign the execute permission to the user.

Any Questions?

Discuss it here: 7 Chmod Command Examples for Beginners

Additional chmod Examples:

Beginners Guide to File and Directory Permissions
(umask, chmod, read, write, execute)

Hack 25. View Multiple Log Files in One Terminal

Typically you may open multiple terminals to view tail -f of multiple files.

For example, if you want to view Apache error_log and access_log at the same time you may do the following in two different terminals.

On one terminal:

```
$ tail -f error_log
```

On another terminal:

```
$ tail -f access_log
```

But, wait! Wouldn't it be nice if you can execute multiple unix tail command in single terminal using one of the following methods?

```
$ multi-tail.sh error_log access_log

(or)

$ tail -f /var/log/syslog -f /var/log/auth.log
```

This hack explains two methods on how to execute multiple Linux tail -f at the same time in single terminal.

Method 1: Custom Shell Script (with Unix tail command)

Create the multitail.sh as shown below.

```
$ vi multi-tail.sh
#!/bin/sh

# When this exits, exit all back ground process also.
trap 'kill $(jobs -p)' EXIT

# iterate through the each given file names,
for file in "$@"
do
```

```
    # show tails of each in background.
    tail -f $file &
done

# wait .. until CTRL+C
wait
```

Now, open multiple files using this new shell script as shown below.

```
$ ./multi-tail.sh error_log access_log
```

Method 2: Standard Linux tail command

The latest version of the Unix tail command supports multiple -f as shown below.

```
$ tail -f /var/log/syslog -f /var/log/auth.log
```

Any Questions?

Discuss it here: 3 Methods To View tail -f output of Multiple Log Files in One Terminal

Hack 26. Less Command

I personally prefer to use less command to view files (instead of opening the file to view in an editor).

Less is similar to more command, but less allows both forward and backward movements. Moreover, less don't require to load the whole file before viewing. Try opening a large log file in Vim editor and less — you'll see the speed difference.

The navigation keys in less command are similar to Vim editor. This hack explains less command navigation and other operations which will make you a better command line warrior.

Search Navigation

Once you've opened a log file (or any file) using less file-name, use the following keys to search. Please note that the match will be highlighted automatically by default.

Use the following shortcut for the less command forward search :

- / – search for a pattern which will take you to the next occurrence.
- n – for next match in forward
- N – for previous match in backward

Use the following shortcut for the less command backward search :

- ? – search for a pattern which will take you to the previous occurrence.
- n – for next match in backward direction
- N – for previous match in forward direction

Tip: If you don't bother about which direction the search is happening, and you want to search file path, or URL, such as "/home/ramesh/", you can use backward search (?pattern) which will be handy as you don't want to escape slashes each time.

```
Search Path
In forward: /\/home\/ramesh\/
In backward: ?/home/ramesh/
```

Screen Navigation

Use the following screen navigation commands while viewing large log files.

- CTRL+F – forward one window

- CTRL+B – backward one window
- CTRL+D – forward half window
- CTRL+U – backward half window

Line navigation

In a smaller chunk of data, where you want to locate particular error, you may want to navigate line by line using these keys:

- j – navigate forward by one line
- k – navigate backward by one line

Other Navigations

The following are other navigation operations that you can use inside the less pager.

- G – go to the end of file
- g – go to the start of file
- q or ZZ – exit the less pager

Simulate tail -f inside less pager – Press F

Once you've opened a file using less command, any content that is appended to the file after that will not be displayed automatically. However, you can press F less command will show the status 'waiting for data'. This is as similar to 'tail -f'.

Count magic

Similar to Vim editor navigation command, you can give 10j to scroll 10 lines down, or 10k to go up by 10 lines.

- 10j – 10 lines forward.
- 10k – 10 lines backward.
- CTRL+G – show the current file name along with line, byte and percentage statistics.

Other useful Less Command Operations

- v – using the configured editor edit the current file.
- h – summary of less commands
- &pattern – display only the matching lines, not all.

Marked navigation

When you are viewing a large log file using less command, you can mark a particular position and return back to that place again by using that mark.

- ma – mark the current position with the letter ‘a’,
- ‘a – go to the marked position ‘a’.

Multiple file paging

Method 1: You can open multiple files by passing the file names as arguments.

```
$ less file1 file2
```

Method 2: While you are viewing file1, use :e to open the file2 as shown below.

$ less file1

```
:e file2
```

Navigation across files: When you opened more than two files (for e.g – less *), use the following keys to navigate between files.

- :n – go to the next file.
- :p – go to the previous file.

Any Questions?

Discuss it here: Less Command: 10 Tips for Effective Navigation

Additional Less Examples:

Open & View 10 Different File Types with Linux Less Command

Hack 27. Wget Examples

wget utility is the best option to download files from internet. wget can pretty much handle all complex download situations including large file downloads, recursive downloads, non-interactive downloads, multiple file downloads etc.,

This hack explains how to use wget for various download scenarios using 15 awesome wget examples.

Download Single File with wget

The following example downloads a single file from internet and stores in the current directory.

```
$ wget http://www.openss7.org/repos/tarballs/strx25-
0.9.2.1.tar.bz2
```

While downloading it will show a progress bar with the following information:

- %age of download completion (for e.g. 31% as shown below)
- Total amount of bytes downloaded so far (for e.g. 1,213,592 bytes as shown below)
- Current download speed (for e.g. 68.2K/s as shown below)
- Remaining time to download (for e.g. eta 34 seconds as shown below)

Download in progress:

```
$ wget http://www.openss7.org/repos/tarballs/strx25-
0.9.2.1.tar.bz2
Saving to: `strx25-0.9.2.1.tar.bz2.1'

31% [=================> 1,213,592   68.2K/s  eta 34s
```

Download completed:

```
$ wget http://www.openss7.org/repos/tarballs/strx25-
0.9.2.1.tar.bz2
Saving to: `strx25-0.9.2.1.tar.bz2'

100%[======================>] 3,852,374   76.8K/s   in 55s

2009-09-25 11:15:30 (68.7 KB/s) - `strx25-0.9.2.1.tar.bz2'
saved [3852374/3852374]
```

Download and Store With a Different File name Using wget -O

By default wget will pick the filename from the last word after last forward slash, which may not be appropriate always.

Wrong: Following example will download and store the file with name: download_script.php?src_id=7701

```
$ wget http://www.vim.org/scripts/download_script.php?
src_id=7701
```

Even though the downloaded file is in zip format, it will get stored in the file as shown below.

```
$ ls
download_script.php?src_id=7701
```

Correct: To correct this issue, we can specify the output file name using the -O option as:

```
$ wget -O taglist.zip
http://www.vim.org/scripts/download_script.php?src_id=7701
```

Specify Download Speed / Download Rate Using wget –limit-rate

While executing the wget, by default it will try to occupy full possible bandwidth. This might not be acceptable when you are downloading huge files on production servers. So, to avoid that we can limit the download speed using the –limit-rate as shown below.

In the following example, the download speed is limited to 200k

```
$ wget --limit-rate=200k
http://www.openss7.org/repos/tarballs/strx25-
0.9.2.1.tar.bz2
```

Continue the Incomplete Download Using wget -c

Restart a download which got stopped in the middle using wget -c option as shown below.

```
$ wget -c http://www.openss7.org/repos/tarballs/strx25-
0.9.2.1.tar.bz2
```

This is very helpful when you have initiated a very big file download which got interrupted in the middle. Instead of starting the whole download again, you can start the download from where it got interrupted using option -c

Note: If a download is stopped in middle, when you restart the download again without the option -c, wget will append .1 to the filename automatically as a file with the previous name already exist. If a file with .1 already exist, it will download the file with .2 at the end.

Download in the Background Using wget -b

For a huge download, put the download in background using wget option -b as shown below.

```
$ wget -b http://www.openss7.org/repos/tarballs/strx25-0.9.2.1.tar.bz2
Continuing in background, pid 1984.
Output will be written to `wget-log'.
```

It will initiate the download and gives back the shell prompt to you. You can always check the status of the download using tail -f as shown below.

```
$ tail -f wget-log
Saving to: `strx25-0.9.2.1.tar.bz2.4'
   0K ..... ........ ......... ..........  1% 65.5K 57s
   150K ......... ............ ..........  5% 86.6K 45s
   250K ......... ............ ..........  7%  182M 46s
   300K ......... ............ ..........  9% 57.9K 47s
```

Mask User Agent and Display wget like Browser Using wget –user-agent

Some websites can disallow you to download its page by identifying that the user agent is not a browser. So you can mask the user agent by using –user-agent options and show wget like a browser as shown below.

```
 $ wget --user-agent="Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.9.0.3) Gecko/2008092416 Firefox/3.0.3" URL-TO-DOWNLOAD
```

Test Download URL Using wget –spider

When you are going to do scheduled download, you should check whether download will happen fine or not at scheduled time. To do so, copy the line exactly from the schedule, and then add –spider option to check.

```
$ wget --spider DOWNLOAD-URL
```

If the URL given is correct, it will say

```
$ wget --spider download-url
Spider mode enabled. Check if remote file exists.
HTTP request sent, awaiting response... 200 OK
Length: unspecified [text/html]
Remote file exists and could contain further links,
but recursion is disabled -- not retrieving.
```

This ensures that the downloading will get success at the scheduled time. But when you had give a wrong URL, you will get the following error.

```
$ wget --spider download-url
Spider mode enabled. Check if remote file exists.
HTTP request sent, awaiting response... 404 Not Found
Remote file does not exist -- broken link!!!
```

You can use the spider option under following scenarios:

- Check before scheduling a download.
- Monitoring whether a website is available or not at certain intervals.
- Check a list of pages from your bookmark, and find out which pages are still exists.

Increase Total Number of Retry Attempts Using wget –tries

If the internet connection has problem, and if the download file is large there is a chance of failures in the download. By default wget retries 20 times to make the download successful.

If needed, you can increase retry attempts using –tries option as shown below.

```
$ wget --tries=75 DOWNLOAD-URL
```

Download Multiple Files / URLs Using Wget -i

First, store all the download files or URLs in a text file as:

```
$ cat > download-file-list.txt
URL1
URL2
URL3
URL4
```

Next, give the download-file-list.txt as argument to wget using -i option as shown below.

```
$ wget -i download-file-list.txt
```

Download a Full Website Using wget –mirror

Following is the command line which you want to execute when you want to download a full website and made available for local viewing.

```
$ wget --mirror -p --convert-links -P ./LOCAL-DIR WEBSITE-URL
```

- –mirror : turn on options suitable for mirroring.
- -p : download all files that are necessary to properly display a given HTML page.
- –convert-links : after the download, convert the links in document for local viewing.
- -P ./LOCAL-DIR : save all the files and directories to the specified directory.

Reject Certain File Types while Downloading Using wget –reject

You have found a website which is useful, but don't want to download the images you can specify the following.

```
$ wget --reject=gif WEBSITE-TO-BE-DOWNLOADED
```

Log messages to a log file instead of stderr Using wget -o

When you wanted the log to be redirected to a log file instead of the terminal.

```
$ wget -o download.log DOWNLOAD-URL
```

Quit Downloading When it Exceeds Certain Size Using wget -Q

When you want to stop download when it crosses 5 MB you can use the following wget command line.

```
$ wget -Q5m -i FILE-WHICH-HAS-URLS
```

Note: This quota will not get effect when you do a download a single URL. That is irrespective of the quota size everything will get downloaded when you specify a single file. This quota is applicable only for recursive downloads.

Download Only Certain File Types Using wget -r -A

You can use this under following situations:

- Download all images from a website
- Download all videos from a website
- Download all PDF files from a website

```
$ wget -r -A.pdf http://url-to-webpage-with-pdfs/
```

FTP Download With wget

You can use wget to perform FTP download as shown below.

Anonymous FTP download using Wget

```
$ wget ftp-url
```

FTP download using wget with username and password authentication.

```
$ wget --ftp-user=USERNAME --ftp-password=PASSWORD
DOWNLOAD-URL
```

Any Questions?

Discuss it here: Wget Download Guide With 15 Examples

Chapter 3: SSH Commands and Tips

Hack 28. Debug SSH Client Session

Sometimes it is necessary to view debug messages to troubleshoot any SSH connection issues. pass -v (lowercase v) option to the ssh as shown below to view the ssh debug messages.

Example without SSH client debug message:

```
$ ssh -l jsmith remotehost.example.com
warning: Connecting to remotehost.example.com failed: No
address associated to the name
```

Example with SSH client debug message:

```
$ ssh -v -l jsmith remotehost.example.com
debug: SshConfig/sshconfig.c:2838/ssh2_parse_config_ext:
Metaconfig parsing stopped at line 3.


debug:
SshConfig/sshconfig.c:637/ssh_config_set_param_verbose:
Setting variable 'VerboseMode' to 'FALSE'.


debug:
SshConfig/sshconfig.c:3130/ssh_config_read_file_ext: Read
17 params from config file.


debug: Ssh2/ssh2.c:1707/main: User config file not found,
using defaults. (Looked for
'/home/jsmith/.ssh2/ssh2_config')


debug: Connecting to remotehost.example.com, port 22…
(SOCKS not used)
```

```
warning: Connecting to remotehost.example.com failed: No
address associated to the name
```

Hack 29. Toggle SSH Session using SSH Escape Character

When you've logged on to the remotehost using ssh from the localhost, you may want to come back to the localhost to perform some activity and go back to remote host again. In this case, you don't need to disconnect the ssh session to the remote host. Instead, follow the steps below.

1. Login to remotehost from localhost:

```
localhost$ ssh -l jsmith remotehost
```

2. Now you are connected to the remotehost:

```
remotehost$
```

3. To come back to the localhost temporarily, type the escape character ~ and Control-Z.

When you type ~ you will not see that immediately on the screen until you press <Control-Z> and press enter. So, on the remotehost in a new line enter the following key strokes for the below to work: ~<Control-Z>

```
remotehost$ ~^Z
[1]+  Stopped  ssh -l jsmith remotehost


localhost$
```

4. Now you are back to the localhost and the ssh remotehost client session runs as a typical UNIX background job, which you can check as shown below:

```
localhost$ jobs
[1]+  Stopped  ssh -l jsmith remotehost
```

5. You can go back to the remote host ssh without entering the password again by bringing the background ssh remotehost session job to foreground on the localhost.

```
localhost$ fg %1
ssh -l jsmith remotehost

remotehost$
```

Hack 30. Display SSH Session Statistics

To get some useful statistics about the current ssh session, do the following. This works only on SSH2 client.

1. Login to remotehost from localhost.

```
localhost$ ssh -l jsmith remotehost
```

2. On the remotehost, type ssh escape character ~ followed by s as shown below. This will display lot of useful statistics about the current SSH connection.

```
remotehost$  [Note: The ~s is not visible on the command
line when you type.]

        remote host: remotehost
        local host: localhost
        remote version: SSH-1.99-OpenSSH_3.9p1
        local version:  SSH-2.0-3.2.9.1 SSH Secure Shell
        compressed bytes in: 1506
        uncompressed bytes in: 1622
```

```
        compressed bytes out: 4997
        uncompressed bytes out: 5118
        packets in: 15
        packets out: 24
        rekeys: 0
        Algorithms:
        Chosen key exchange algorithm: diffie-hellman-
group1-sha1
        Chosen host key algorithm: ssh-dss
        Common host key algorithms: ssh-dss,ssh-rsa
        Algorithms client to server:
        Cipher: aes128-cbc
        MAC: hmac-sha1
        Compression: zlib
        Algorithms server to client:
        Cipher: aes128-cbc
        MAC: hmac-sha1
        Compression: zlib

localhost$
```

Any questions?

Discuss it here: SSH Client commands

Hack 31. Change OpenSSH Security Options

OpenSSH options are controlled through the /etc/ssh/sshd_config file. This hack explains the 7 default options in sshd_config file that you should change.

In sshd_config, the lines that start with # are comments. For those options that uses the default values, the sshd_config file contains a commented line with the option and its default value.

This makes it easier for us, as we can see the OpenSSH option name and the default value without having to lookup somewhere else.

For example, sshd_config file contains the following commented line. This indicates that the PubkeyAuthentication option contains “yes” as the default value.

```
$ grep -i pubkey /etc/ssh/sshd_config
#PubkeyAuthentication yes
```

If you like to change this, you should remove the comment and change the value (from yes to no) as shown below.

```
$ vi /etc/ssh/sshd_config
PubkeyAuthentication no
```

I showed the above only as an example. You don’t need to change the default value of PubkeyAuthentication option, as allowing public key authentication is good.

You don’t need to modify any of the default values in the sshd_config file except the 7 options mentioned in this hack.

Disable Root Login (PermitRootLogin)

By default you can ssh to the server as root. It is best not to allow root to login directly to the server. Instead, you should login to the system as your account and then do ‘su -’ to login as root.

If you have multiple sysadmins in your organization, and if they all login to the server directly as root, you might not know which sysadmin logged in as root. Instead, if you disable login as root, sysadmins are

forced to login as their account first, before they can do ‘su -’, this makes the auditing easier.

Add the following entry to sshd_config to disable root to login to the server directly.

```
$ vi /etc/ssh/sshd_config
PermitRootLogin no
```

Allow Only Specific Users or Groups (AllowUsers AllowGroups)

By default anybody who is authenticated successfully are allowed to login. Instead you can restrict which users (or groups) you allow to login to the system.

This is helpful when you have created several user accounts on the system, but want only few of them to login.

This is also helpful when you are using NIS, openLDAP (or some other external system) for authentication. Every user in your company might have account on NIS, OpenLDAP etc. But, on a specific server you want only few of them to login. For example, on production system you want only sysadmins to login.

Add the following entry to the sshd_config file to allow only specific users to login to the system. In the example below only ramesh, john and jason can login to this system. Usernames should be separated by space.

```
$ vi /etc/ssh/sshd_config
AllowUsers ramesh john jason
```

Add the following entry to the sshd_config file to allow only the users who belong to a specific group to login. In the example below only users who belong to sysadmin and dba group can login to the system.

```
$ vi /etc/ssh/sshd_config
```

```
AllowGroups sysadmin dba
```

Deny Specific Users or Groups (DenyUsers DenyGroups)

Instead of allowing specific users (or groups), you can also deny specific users or groups.

Add the following entry to the sshd_config file to deny specific users to login to the system. In the example below cvs, apache, jane cannot login to this system. Usernames should be separated by space.

```
$ vi /etc/ssh/sshd_config
DenyUsers cvs apache jane
```

Add the following entry to the sshd_config file to deny users who belong to a specific group to login. In the example below users who belong to developers and qa group cannot login to the system.

```
$ vi /etc/ssh/sshd_config
DenyGroups developers qa
```

Note: You can use combination of all the Allow and Deny directives. It is processed in this order: DenyUsers, AllowUsers, DenyGroups, and finally AllowGroups

Change SSHD Port Number (Port)

By default ssh runs on port 22. Most of the attackers will check if a server is open on port 22, and will randomly use brute force to login to the server using several username and password combination.

If you change the port # to something different, others need to know exactly what port to use to login to the server using ssh. The example below uses port 222 for ssh.

```
$ vi /etc/ssh/sshd_config
Port 222
```

From your logs (/var/log/secure), if you see lot of invalid logins using ssh for accounts that don't exist on your system, from the ip-address that you don't recognize, it might be some brute-force attack. Those kind of ssh invalid login will stop, if you change the port number.

Please note that this causes little inconvenience to your team who login to the system, as they need to know both the ip-address and the port number.

Change Login Grace Time (LoginGraceTime)

When you ssh to a server, you have 2 minutes to login. If you don't successfully login within 2 minutes, ssh will disconnect.

2 minutes time to login successfully is too much. You should consider changing it to 30 seconds, or may be 1 minute.

Add the following entry to the sshd_config file to change the login grace time from 2 minutes to 1 minute.

```
$ vi /etc/ssh/sshd_config
LoginGraceTime 1m
```

Restrict the Interface (IP Address) to Login (ListenAddress)

If you have multiple interfaces on the server that are configured to different ip-address, you might not want everybody to login to the server using all those ip-address.

Let us assume that you have the following 4 interfaces on the server:

- eth0 – 192.168.10.200
- eth1 – 192.168.10.201
- eth2 – 192.168.10.202
- eth3 – 192.168.10.203

By default ssh will listen on all of the above ip-addresses. If you want users to login only using ip-address 200 and 202, do the following in your sshd_config

```
$ vi /etc/ssh/sshd_config
ListenAddress 192.168.10.200
ListenAddress 192.168.10.202
```

Disconnect SSH when no activity (ClientAliveInterval)

Once you've successfully logged in to the system, you might want to get disconnected when there are no activities after x number of minutes. This is basically idle timeout.

In Bash, you can achieve this using TMOUT variable.

In OpenSSH, this can be achieved by combining ClientAliveCountMax and ClientAliveInterval options in sshd_config file.

- ClientAliveCountMax – This indicates the total number of checkalive message sent by the ssh server without getting any response from the ssh client. Default is 3.
- ClientAliveInterval – This indicates the timeout in seconds. After x number of seconds, ssh server will send a message to the client asking for response. Deafult is 0 (server will not send message to client to check.).

If you want ssh client to exit (timeout) automatically after 10 minutes (600 seconds), modify the sshd_config file and set the following two parameters as shown below.

```
$ vi /etc/ssh/sshd_config
ClientAliveInterval 600
ClientAliveCountMax 0
```

Any Questions?

Discuss it here: 7 Default OpenSSH Security Options that You Should Change

Hack 32. Transfer All PuTTY Sessions

If you want to connect from a Windows machine to a Linux server, you need to use one of the SSH clients. PuTTY is hands-down the best SSH client for Windows. It's light-weight with a single putty.exe file and nothing else to install.

If you have not used PuTTY earlier, you may want to download PuTTY software from the official PuTTY download page before exploring these PuTTY tricks.

Move Putty sessions to another computer

PuTTY stores all the session information in Windows registry. On the source machine, export the PuTTY session registry information as shown below.

```
C:> regedit /e "%userprofile%\desktop\putty-registry.reg"
HKEY_CURRENT_USER\Software\Simontatham
```

Transfer this putty-registry.reg file from source machine to destination machine.

On the destination machine, import the PuTTY SSH session registry, by right mouse-click on the putty-registry.reg and click on 'Merge'. This will transfer all the PuTTY session information from one windows system to another.

Delete All PuTTY Sessions Together

When you are swapping an old computer with a new computer, you may end-up transferring all PuTTY sessions to new computer. Once you've

transferred all PuTTY sessions, execute “putty -cleanup” from the command line as shown below.

```
C:>putty -cleanup
```

This will display a warning message. Click on ‘Yes’ to wipe-out all PuTTY session and random seed files from the Windows registry. This is a better process to delete all saved PuTTY sessions from Windows registry instead of deleting one-by-one from the PuTTY session list.

More PuTTY Tips:

Turbocharge PuTTY with 12 Powerful Add-Ons

10 Awesome PuTTY Tips and Tricks You Probably Didn’t Know

Extreme Makeover Using PuTTY Connection Manager

Chapter 4: Date Manipulation

Hack 33. Set System Date and Time

To change the system date use:

```
# date {mmddhhmiyyyy.ss}
```

- mm – Month
- dd – Date
- hh – 24 hour format
- mi – Minutes
- yyyy – Year
- ss – seconds

For example, to set system date to Jan 31st 2009, 10:19 p.m, 53 seconds

```
# date 013122192009.53
```

You can also change system date using set argument as shown below.

```
# date 013122192009.53

# date +%Y%m%d -s "20090131"

# date -s "01/31/2009 22:19:53"

# date -s "31 JAN 2009 22:19:53"

# date set="31 JAN 2009 22:19:53"
```

To set the time only:

```
# date +%T -s "22:19:53"

# date +%T%p -s "10:19:53PM"
```

Hack 34. Set Hardware Date and Time

Before setting the hardware date and time, make sure the OS date and time is set appropriately as shown in previous hack.

Set the hardware date and time based on the system date as shown below:

```
# hwclock –systohc

# hwclock --systohc –utc
```

Use hwclock without any parameter, to view the current hardware date and time:

```
# hwclock
```

Check the clock file to verify whether the system is set for UTC:

```
# cat /etc/sysconfig/clock
ZONE="America/Los_Angeles"
UTC=false
ARC=false
```

Hack 35. Display Date and Time in a Specific Format

The following are different ways of displaying the current date and time in various formats:

```
$ date
Thu Jan  1 08:19:23 PST 2009

$ date --date="now"
Thu Jan  1 08:20:05 PST 2009

$ date --date="today"
Thu Jan  1 08:20:12 PST 2009

$ date --date='1970-01-01 00:00:01 UTC +5 hours' +%s
18001

$ date '+Current Date: %m/%d/%y%nCurrent Time:%H:%M:%S'
Current Date: 01/01/09
Current Time:08:21:41

$ date +"%d-%m-%Y"
01-01-2009

$ date +"%d/%m/%Y"
01/01/2009

$ date +"%A,%B %d %Y"
Thursday,January 01 2009
```

The following are the different format options you can pass to the date command:

- %D date (mm/dd/yy)
- %d day of month (01..31)
- %m month (01..12)
- %y last two digits of year (00..99)
- %a locale's abbreviated weekday name (Sun..Sat)
- %A locale's full weekday name, variable length (Sunday..Saturday)
- %b locale's abbreviated month name (Jan..Dec)
- %B locale's full month name, variable length (January..December)
- %H hour (00..23)
- %I hour (01..12)
- %Y year (1970...)

Hack 36. Display Past Date and Time

The following are various ways to display a past date and time:

```
$ date --date='3 seconds ago'
Thu Jan  1 08:27:00 PST 2009

$ date --date="1 day ago"
Wed Dec 31 08:27:13 PST 2008

$ date --date="1 days ago"
Wed Dec 31 08:27:18 PST 2008

$ date --date="1 month ago"
Mon Dec  1 08:27:23 PST 2008
```

```
$ date --date="1 year ago"
Tue Jan  1 08:27:28 PST 2008

$ date --date="yesterday"
Wed Dec 31 08:27:34 PST 2008

$ date --date="10 months 2 day ago"
Thu Feb 28 08:27:41 PST 2008
```

Hack 37. Display Future Date and Time

The following examples shows how to display a future date and time.

```
$ date
Thu Jan  1 08:30:07 PST 2009

$ date --date='3 seconds'
Thu Jan  1 08:30:12 PST 2009

$ date --date='4 hours'
Thu Jan  1 12:30:17 PST 2009

$ date --date='tomorrow'
Fri Jan  2 08:30:25 PST 2009

$ date --date="1 day"
Fri Jan  2 08:30:31 PST 2009

$ date --date="1 days"
Fri Jan  2 08:30:38 PST 2009

$ date --date="2 days"
```

```
Sat Jan  3 08:30:43 PST 2009

$ date --date='1 month'
Sun Feb  1 08:30:48 PST 2009

$ date --date='1 week'
Thu Jan  8 08:30:53 PST 2009

$ date --date="2 months"
Sun Mar  1 08:30:58 PST 2009

$ date --date="2 years"
Sat Jan  1 08:31:03 PST 2011

$ date --date="next day"
Fri Jan  2 08:31:10 PST 2009

$ date --date="-1 days ago"
Fri Jan  2 08:31:15 PST 2009

$ date --date="this Wednesday"
Wed Jan  7 00:00:00 PST 2009
```

Chapter 5: PS1, PS2, PS3, PS4 and PROMPT_COMMAND

Hack 38. PS1 - Default Interaction Prompt

The default interactive prompt on your Linux can be modified as shown below to something useful and informative. In the following example, the default PS1 was "\s-\v\$", which displays the shell name and the version number. Let us change this default behavior to display the username, hostname and current working directory name as shown below.

```
-bash-3.2$  export PS1="\u@\h \w> "

ramesh@dev-db ~> cd /etc/mail

ramesh@dev-db /etc/mail>

[Note: Prompt changed to "username@hostname current-dir>"
format]
```

Following PS1 codes are used in this example:

- \u – Username
- \h – Hostname
- \w - Full pathname of current directory. Please note that when you are in the home directory, this will display only ~ as shown above

Note that there is a space at the end in the value of PS1. Personally, I prefer a space at the end of the prompt for better readability.

Make this setting permanent by adding export PS1="\u@\h \w> " to either .bash_profile (or) .bashrc as shown below.

```
ramesh@dev-db ~> vi ~/.bash_profile


ramesh@dev-db ~> vi ~/.bashrc


[Note: Add export PS1="\u@\h \w> " to one of the above
files]
```

Hack 39. PS2 - Continuation Interactive Prompt

A very long command can be broken down to multiple lines by giving \ at the end of the line. The default interactive prompt for a multi-line command is "> ". Let us change this default behavior to display "continue->" by using PS2 environment variable as shown below.

```
ramesh@dev-db ~> myisamchk --silent --force --fast
--update-state \
> --key_buffer_size=512M --sort_buffer_size=512M \
> --read_buffer_size=4M --write_buffer_size=4M \
> /var/lib/mysql/bugs/*.MYI


[Note: This uses the default ">" for continuation prompt]


ramesh@dev-db ~> export PS2="continue-> "


ramesh@dev-db ~> myisamchk --silent --force --fast
--update-state \
continue-> --key_buffer_size=512M
--sort_buffer_size=512M \
continue-> --read_buffer_size=4M --write_buffer_size=4M \
continue-> /var/lib/mysql/bugs/*.MYI


[Note: This uses the modified "continue-> " for
continuation prompt]
```

I found it very helpful and easy to read, when I break my long commands into multiple lines using \. I have also seen others who don't like to break-up long commands.

Hack 40. PS3 - Prompt Used by Select Command

You can define a custom prompt for the select loop inside a shell script, using the PS3 environment variable, as explained below.

Shell script and output WITHOUT PS3:

```
ramesh@dev-db ~> cat ps3.sh
select i in mon tue wed exit
do
  case $i in
    mon) echo "Monday";;
    tue) echo "Tuesday";;
    wed) echo "Wednesday";;
    exit) exit;;
  esac
done

ramesh@dev-db ~> ./ps3.sh
1) mon
2) tue
3) wed
4) exit
#? 1
Monday
#? 4
```

```
[Note: This displays the default "#?" for select command
prompt]
```

Shell script and output WITH PS3:

```
ramesh@dev-db ~> cat ps3.sh
PS3="Select a day (1-4): "
select i in mon tue wed exit
do
  case $i in
    mon) echo "Monday";;
    tue) echo "Tuesday";;
    wed) echo "Wednesday";;
    exit) exit;;
  esac
done

ramesh@dev-db ~> ./ps3.sh
1) mon
2) tue
3) wed
4) exit
Select a day (1-4): 1
Monday
Select a day (1-4): 4

[Note: This displays the modified "Select a day (1-4):"
for select command prompt]
```

Hack 41. PS4 - Prompt to Prefix Tracing Output

The PS4 shell variable defines the prompt that gets displayed, when you execute a shell script in debug mode (using set -x) as shown below.

Shell script and output WITHOUT PS4:

```
ramesh@dev-db ~> cat ps4.sh
set -x
echo "PS4 demo script"
ls -l /etc/ | wc -l
du -sh ~

ramesh@dev-db ~> ./ps4.sh
       ++ echo 'PS4 demo script'
PS4 demo script
++ ls -l /etc/
++ wc -l
243
++ du -sh /home/ramesh
48K     /home/ramesh

[Note: This displays the default "++" while tracing the
output using set -x]
```

Shell script and output WITH PS4:

The PS4 defined below in the ps4.sh has the following two codes:

- $0 - indicates the name of script
- $LINENO - displays the current line number within the script

```
ramesh@dev-db ~> cat ps4.sh
```

```
export PS4='$0.$LINENO+ '
set -x
echo "PS4 demo script"
ls -l /etc/ | wc -l
du -sh ~

ramesh@dev-db ~> ./ps4.sh
../ps4.sh.3+ echo 'PS4 demo script'
PS4 demo script
../ps4.sh.4+ ls -l /etc/
../ps4.sh.4+ wc -l
243
../ps4.sh.5+ du -sh /home/ramesh
48K     /home/ramesh

[Note: This displays the modified "{script-name}.{line-
number}+" while tracing the output using set -x]
```

Hack 42. PROMPT_COMMAND

Bash shell executes the content of the PROMPT_COMMAND just before displaying the PS1 variable.

```
ramesh@dev-db ~> export PROMPT_COMMAND="date +%H:%M:%S"


22:08:42
ramesh@dev-db ~>
[Note: This displays the PROMPT_COMMAND and PS1 output on
different lines]
```

If you want to display the value of PROMPT_COMMAND in the same line as the PS1, use the echo -n as shown below.

```
ramesh@dev-db ~> export PROMPT_COMMAND="echo -n [$(date +
%H:%M:%S)]"


[22:08:51]ramesh@dev-db ~>


[Note: This displays the PROMPT_COMMAND and PS1 output on
the same line]
```

Any Questions?

Discuss it here: Bash Shell: Take Control of PS1, PS2, PS3, PS4 and PROMPT_COMMAND

Hack 43. Customize Bash Prompt Using PS1

Display username, hostname and basename of directory in the prompt

The PS1 in this example displays following three information in the prompt:

- \u – Username
- \h – Hostname
- \W – Base name of the current working directory

```
-bash-3.2$ export PS1="\u@\h \W> "


ramesh@dev-db ~> cd /etc/mail


ramesh@dev-db mail>
```

Display current time in the prompt

In the PS1 environment variable, you can directly execute any Linux command, by specifying in the format $(linux_command). In the following example, the command $(date) is executed to display the current time inside the prompt.

```
ramesh@dev-db ~> export PS1="\u@\h [\$(date +%H:%M:%S)]> "

ramesh@dev-db [11:09:56]>
```

You can also use \t to display the current time in the hh:mm:ss format as shown below:

```
ramesh@dev-db ~> export PS1="\u@\h [\t]> "

ramesh@dev-db [12:42:55]>
```

You can also use \@ to display the current time in 12-hour am/pm format as shown below:

```
ramesh@dev-db ~> export PS1="[\@] \u@\h> "

[04:12 PM] ramesh@dev-db>
```

Display output of any command in the prompt

You can display output of any Linux command in the prompt. The following example displays three items separated by | (pipe) in the command prompt:

- \!: The history number of the command
- \h: hostname
- $kernel_version: The output of the uname -r command from $kernel_version variable
- \$?: Status of the last command

```
ramesh@dev-db ~> kernel_version=$(uname -r)
```

```
ramesh@dev-db ~> export PS1="\!|\h|$kernel_version|\$?> "

473|dev-db|2.6.25-14.fc9.i686|0>
```

Create your own prompt using the available codes for PS1 variable

Use the following codes and create your own personal PS1 Linux prompt that is functional and suites your taste.

- \a an ASCII bell character (07)
- \d the date in “Weekday Month Date” format (e.g., “Tue May 26″)
- \D{format} - the format is passed to strftime(3) and the result is inserted into the prompt string; an empty format results in a locale-specific time representation. The braces are required
- \e an ASCII escape character (033)
- \h the hostname up to the first part
- \H the hostname
- \j the number of jobs currently managed by the shell
- \l the basename of the shell’s terminal device name
- \n newline
- \r carriage return
- \s the name of the shell, the basename of $0 (the portion following the final slash)
- \t the current time in 24-hour HH:MM:SS format
- \T the current time in 12-hour HH:MM:SS format
- \@ the current time in 12-hour am/pm format
- \A the current time in 24-hour HH:MM format
- \u the username of the current user
- \v the version of bash (e.g., 2.00)
- \V the release of bash, version + patch level (e.g., 2.00.0)

- \w the current working directory, with $HOME abbreviated with a tilde
- \W the basename of the current working directory, with $HOME abbreviated with a tilde
- \! the history number of this command
- \# the command number of this command
- \$ if the effective UID is 0, a #, otherwise a $
- \nnn the character corresponding to the octal number nnn
- \\ a backslash
- \[begin a sequence of non-printing characters, which could be used to embed a terminal control sequence into the prompt
- \] end a sequence of non-printing character

Use bash shell function inside PS1 variable

You can also invoke a bash shell function in the PS1 as shown below.

```
ramesh@dev-db ~> function httpdcount {
>  ps aux | grep httpd | grep -v grep | wc -l
> }

ramesh@dev-db ~> export PS1="\u@\h [`httpdcount`]> "

ramesh@dev-db [12]>

[Note: This displays the total number of running httpd
processes]
```

You can add the following line to your ~/.bash_profile or ~/.bashrc to make this change permanent:

```
$ vi .bash_profile
function httpdcount {
  ps aux | grep httpd | grep -v grep | wc -l
```

```
}
export PS1='\u@\h [`httpdcount`]> '
```

Note: You can also use “pgrep httpd | wc –l” instead of the “ps aux | grep httpd | grep -v grep | wc –l” in the above httpdcount function.

Use shell script inside PS1 variable

You can also invoke a shell script inside the PS1 variable. In the example below, the ~/bin/totalfilesize.sh, which calculates the total filesize of the current directory, is invoked inside the PS1 variable.

```
ramesh@dev-db ~> cat ~/bin/totalfilesize.sh

for filesize in $(ls -l . | grep "^-" | awk '{print $5}')
do
  let totalsize=$totalsize+$filesize
done
echo -n "$totalsize"

ramesh@dev-db ~> export PATH=$PATH:~/bin

ramesh@dev-db ~> export PS1="\u@\h [\$(totalfilesize.sh)
bytes]> "

ramesh@dev-db [534 bytes]> cd /etc/mail

ramesh@dev-db [167997 bytes]>

[Note: This executes the totalfilesize.sh to display the
total file size of the current directory in the PS1
prompt]
```

You can also write the ~/bin/totalfilesize.sh as shown below without the for loop.

```
ramesh@dev-db ~> cat ~/bin/totalfilesize.sh

ls -l | awk '/^-/ { sum+=$5 } END { printf sum }'
```

Hack 44. Colorful Bash Prompt Using PS1

Change foreground color of the prompt

Display prompt in blue color, along with username, host and current directory information .

Use the following is for light blue prompt:

```
$ export PS1="\e[0;34m\u@\h \w> \e[m "
```

Use the following for dark blue prompt:

```
$ export PS1="\e[1;34m\u@\h \w> \e[m "
```

- \e[- Indicates the beginning of color prompt
- x;ym - Indicates color code. Use the color code values mentioned below.
- \e[m - indicates the end of color prompt

Color Code Table:

- Black 0;30
- Blue 0;34
- Green 0;32
- Cyan 0;36
- Red 0;31
- Purple 0;35
- Brown 0;33

[Note: Replace 0 with 1 for dark color]

Make the color change permanent by adding the following lines your ~/.bash_profile or ~/.bashrc

```
 $ vi ~/.bash_profile
STARTCOLOR='\e[0;34m';
ENDCOLOR="\e[0m"
export PS1="$STARTCOLOR\u@\h \w> $ENDCOLOR"
```

Change background color of the prompt

Change the background color by specifying \e[{code}m in the PS1 prompt as shown below.

Use the following for Light Gray background:

```
$ export PS1="\e[47m\u@\h \w> \e[m "
```

Combination of background and foreground

Use the following for Light Blue foreground and Light Gray background

```
$ export PS1="\e[0;34m\e[47m\u@\h \w> \e[m "
```

Add the following to your ~/.bash_profile or ~/.bashrc to make the above background and foreground color permanent.

```
$ vi ~/.bash_profile
STARTFGCOLOR='\e[0;34m';
STARTBGCOLOR="\e[47m"
ENDCOLOR="\e[0m"
export PS1="$STARTFGCOLOR$STARTBGCOLOR\u@\h \w> $ENDCOLOR"
```

Play around by using the following background color and choose the one that match your taste:

- \e[40m
- \e[41m
- \e[42m
- \e[43m
- \e[44m
- \e[45m
- \e[46m
- \e[47m

Display multiple colors in the prompt

You can also display multiple colors in the same prompt. Add the following function to your ~/.bash_profile

```
function prompt {
  local BLUE="\[\033[0;34m\]"
  local DARK_BLUE="\[\033[1;34m\]”
  local RED=”\[\033[0;31m\]”
  local DARK_RED=”\[\033[1;31m\]”
  local NO_COLOR=”\[\033[0m\]”
  case $TERM in
    xterm*|rxvt*)
      TITLEBAR=’\[\033]0;\u@\h:\w\007\]’
      ;;
    *)
      TITLEBAR=””
      ;;
  esac
  PS1=”\u@\h [\t]> ”
  PS1=”${TITLEBAR}\
  $BLUE\u@\h $RED[\t]>$NO_COLOR ”
  PS2=’continue-> ‘
  PS4=’$0.$LINENO+ ‘
```

```
}
```

You can re-login for the changes to take effect or source the .bash_profile as shown below.

```
$. ./.bash_profile


$ prompt
ramesh@dev-db [13:02:13]>
```

Change the prompt color using tput

You can also change color of the PS1 prompt using tput as shown below:

```
$ export PS1="\[$(tput bold)$(tput setb 4)$(tput setaf
7)\]\u@\h:\w $ \[$(tput sgr0)\]“
```

tput Color Capabilities:

- tput setab [1-7] - Set a background color using ANSI escape
- tput setb [1-7] - Set a background color
- tput setaf [1-7] - Set a foreground color using ANSI escape
- tput setf [1-7] - Set a foreground color

tput Text Mode Capabilities:

- tput bold - Set bold mode
- tput dim - turn on half-bright mode
- tput smul - begin underline mode
- tput rmul - exit underline mode
- tput rev - Turn on reverse mode
- tput smso - Enter standout mode (bold on rxvt)
- tput rmso - Exit standout mode
- tput sgr0 - Turn off all attributes

Color Code for tput:

- 0 – Black
- 1 – Red
- 2 – Green
- 3 – Yellow
- 4 – Blue
- 5 – Magenta
- 6 – Cyan
- 7 - White

Any Questions?

Discuss it here: Bash Shell PS1: 10 Examples to Make Your Linux Prompt like Angelina Jolie

Additional tput Examples:

9 UNIX / Linux tput Examples: Control Your Terminal Color and Cursor

Chapter 6: Archive and Compression

Hack 45. Zip Command Basics

How to zip multiple files?

```
syntax: zip {.zip file-name} {file-names}
```

```
# zip var-log-files.zip /var/log/*
  adding: var/log/acpid (deflated 81%)
  adding: var/log/anaconda.log (deflated 79%)
  adding: var/log/anaconda.syslog (deflated 73%)
  adding: var/log/anaconda.xlog (deflated 82%)
  adding: var/log/audit/ (stored 0%)
  adding: var/log/boot.log (stored 0%)
  adding: var/log/boot.log.1 (deflated 40%)
  adding: var/log/boot.log.2 (deflated 42%)
  adding: var/log/boot.log.3 (deflated 40%)
  adding: var/log/boot.log.4 (deflated 40%)
```

How to zip a directory and it's files recursively?

```
# zip -r var-log-dir.zip /var/log/
updating: var/log/ (stored 0%)
  adding: var/log/wtmp (deflated 78%)
  adding: var/log/scrollkeeper.log (deflated 94%)
  adding: var/log/rpmpkgs.3 (deflated 68%)
  adding: var/log/spooler (stored 0%)
  adding: var/log/cron.2 (deflated 90%)
```

```
  adding: var/log/spooler.1 (stored 0%)
  adding: var/log/spooler.4 (stored 0%)
  adding: var/log/httpd/ (stored 0%)
  adding: var/log/rpmpkgs.1 (deflated 68%)
  adding: var/log/anaconda.log (deflated 79%)
  adding: var/log/secure.2 (deflated 93%)
```

How to unzip a *.zip compressed file?

```
# unzip var-log.zip
Archive:  var-log.zip
  inflating: var/log/acpid
  inflating: var/log/anaconda.log
  inflating: var/log/anaconda.syslog
  inflating: var/log/anaconda.xlog
   creating: var/log/audit/
```

To see a detailed output during unzip pass the –v option as shown below.

```
# unzip -v var-log.zip

Archive: var-log.zip
Length Method Size Ratio Date Time   CRC-32    Name
--------  ------  ------- -----   ----   ----  ------
1916  Defl:N      369  81%  02-08-08 14:27  e2ffdc0c
var/log/acpid
13546  Defl:N     2900  79%  02-02-07 14:25  34cc03a1
var/log/anaconda.log

skip..

7680  Defl:N      411  95%  12-30-08 10:55  fe876ee9
var/log/wtmp.1
```

```
40981  Defl:N     7395  82%  02-08-08 14:28  6386a95e
var/log/Xorg.0.log
--------          -------  ---
41406991          2809229  93%
56 files
```

How to list a content of zip file with uncompressing it?

```
# unzip -l var-log.zip

Archive:  var-log.zip
  Length     Date   Time    Name
 --------    ----   ----    ----
     1916  02-08-08 14:27   var/log/acpid
    13546  02-02-07 14:25   var/log/anaconda.log

..skip..

    40981  02-08-08 14:28   var/log/Xorg.0.log
    40981  02-08-07 14:56   var/log/Xorg.0.log.old
 --------                   -------
 41406991                   56 files
```

Hack 46. Zip Command Advanced Compression

There are 10 levels of compression provided by zip command.

- Level 0 is the lowest level, where it just archives the file without any compression.
- Level 1 will perform little compression. But, will be very fast.
- Level 6 is the default level of compression.

- Level 9 is the maximum compression. This will be slower when compared to default level. In my opinion, unless you are compressing a huge file, you should always use level 9.

In the example below, I used Level 0, default Level 6, and Level 9 compression on a same directory. See the compressed file size yourself.

```
# zip var-log-files-default.zip /var/log/*

# zip -0 var-log-files-0.zip /var/log/*

# zip -9 var-log-files-9.zip /var/log/*

# ls -ltr
-rw-r--r--   1 root       root        2817248 Jan  1 13:05
var-log-files-default.zip
-rw-r--r--   1 root       root       41415301 Jan  1 13:05
var-log-files-0.zip
-rw-r--r--   1 root       root        2582610 Jan  1 13:06
var-log-files-9.zip
```

Validate a zip archive

Sometime you may want to validate a zip archive without extracting it. To test the validity of the zip file, pass option –t as shown below.

```
# unzip -t var-log.zip

     Archive:  var-log.zip
    testing: var/log/acpid            OK
    testing: var/log/anaconda.log     OK
    testing: var/log/anaconda.syslog   OK

skip...

    testing: var/log/wtmp             OK
```

```
    testing: var/log/wtmp.1           OK
    testing: var/log/Xorg.0.log       OK
```

No errors detected in compressed data of var-log.zip.

Hack 47. Password Protection of Zip files

Pass the option –P to the zip command to assign a password to the zip file.

```
# zip -P mysecurepwd var-log-protected.zip /var/log/*
```

The above option is good if you are using the command inside a shell-script for background jobs. However, when you are performing the compression interactively on the command-line, you don't want the password to be visible in the history. So, use the option –e as shown below to assign the password.

```
# zip -e var-log-protected.zip /var/log/*
Enter password:
Verify password:
updating: var/log/acpid (deflated 81%)
updating: var/log/anaconda.log (deflated 79%)
```

When you are uncompressing a password protected file, it will ask for the password as shown below.

```
# unzip  var-log-protected.zip
Archive:  var-log-protected.zip
[var-log-protected.zip] var/log/acpid password:
```

Hack 48. Tar Command Examples

tar command (tape archive) is used to convert a group of files into an archive.

Syntax:

```
tar [options] [tar-archive-name] [other-file-names]
```

How can I create a single backup file of all files and subdirectories under my home directory?

The following command creates a single archive backup file called my_home_directory.tar under /tmp. This archive will contain all the files and subdirectories under /home/jsmith.

- Option c, stands for create an archive.
- Option v stands for verbose mode, displays additional information while executing the command.
- Option f indicates the archive file name mentioned in the command.

```
# tar cvf /tmp/my_home_directory.tar /home/jsmith
```

How do I view all the files inside the tar archive?

Option t will display all the files from the tar archive.

```
# tar tvf /tmp/my_home_directory.tar
```

How do I extract all the files from a tar archive?

Option x will extract the files from the tar archive as shown below. This will extract the content to the current directory location from where the command is executed.

```
# tar xvf /tmp/my_home_directory.tar
```

How do I extract tar.gz files to a specific directory?

```
# tar xvfz /tmp/my_home_directory.tar.gz –C /home/ramesh
```

Additional Tar Examples:

The Ultimate Tar Command Tutorial with 10 Practical Examples

Hack 49. Combine gzip, bzip2 with Tar

How to use gzip with tar?

Add option z to the tar command when dealing with tar.gz compressed file.

```
# tar cvfz /tmp/my_home_directory.tar.gz /home/jsmith

# tar xvfz /tmp/my_home_directory.tar.gz

# tar tvfz /tmp/my_home_directory.tar.gz
```

Note: Using gzip is faster when compared to bzip2.

How to use bzip2 with tar?

Add option j to the tar command when dealing with tar.bz2 compressed file.

```
# tar cvfj /tmp/my_home_directory.tar.bz2 /home/jsmith

# tar xvfj /tmp/my_home_directory.tar.bz2

# tar tvfj /tmp/my_home_directory.tar.bz2
```

Note: Using bizp2 gives higher level of compression when compared to gzip.

Hack 50. BZ is Eazy! Bz* Command Examples

bzip2 command is used for compression and decompression of files. The main advantage of bzip2 is the best compression size.

bzip2 vs gzip:

- bzip2 will compress better than gzip.
- Speed of bzip2 is somewhat slower than gzip and zip.
- bzip2 provides high rate of compression with reasonable speed.

There are several Linux bz commands available to manipulate the bzip2 files. This hack explains various bz commands with 6 practical examples.

Compressing a file using bzip2

When you compress a file using bzip2 command, it creates a compressed file with *.bz2 extension as shown below.

```
$ bzip2 trace

$ ls -l trace.bz2
-rw-r--r-- 1 root root       54167 Jan 23  2009 trace.bz2
```

Search operation in bzip2 file using bzgrep

bzgrep command is used to search for a string or a pattern (regular expression) on bzip2 compressed files.

bzgrep will apply grep to data from files in the bzip2 format without requiring on-disk decompression. So all the options of a grep command will be applied for bzgrep also.

Syntax:

```
bzgrep grep-options -e pattern filename
```

In the below example, trace.bz2 is a compressed trace file which is of size 58M.

```
$ bzgrep -i "CONSOLE=.*"  trace.bz2
2010-10-11T08:40:28.100 gs(16985): CONSOLE=/dev/pts/0
2010-10-11T08:40:29.772 gs(17031): CONSOLE=/dev/pts/0
2010-10-11T08:40:58.140 gs(17099): CONSOLE=/dev/pts/0
2010-10-11T08:41:27.547 gs(17164): CONSOLE=/dev/pts/0
2010-10-11T08:41:57.962 gs(17233): CONSOLE=/dev/pts/0
2010-10-11T08:42:28.392 gs(17294): CONSOLE=/dev/pts/0
2010-10-11T08:42:57.721 gs(17439): CONSOLE=/dev/pts/0
```

If bzgrep is not there, you have to decompress the file manually and do a grep on that, where bzgrep does this process internally and gives you the required output. bzegrep and bzfgrep commands will apply egrep and fgrep operation on bzip2 files respectively.

View the bzip2 file using bzcat

If you want only to read the compressed file without decompressing it, use the bzcat command as shown below.

```
$ bzcat trace.bz2
0: ERR: Wed Sep 22 09:59:42 2010:
gs(11153/47752677794640): [chk_sqlcode.scp:92]: Database:
ORA-01653: unable to extend table OPC_OP.OP
C_HIST_MESSAGES (OpC50-15)
0: ERR: Wed Sep 22 09:59:47 2010:
gs(11153/47752677794640): [chk_sqlcode.scp:92]: Database:
ORA-01653: unable to extend table OPC_OP.OP
C_HIST_MESSAGES (OpC50-15)
Retry. (OpC51-22)
Database: ORA-01653: unable to extend table
OPC_OP.OPC_HIST_MESSAGES by 64 in tablespace OPC_6
(OpC50-15)
.
```

```
.
```

bzcat command displays the uncompressed content into standard output file for the users to view the content.

Paging bzip2 compressed file with bzless & bzmore

bzless and bzmore command allows you to view the content of bzip2 compressed files page by page. bzmore works on files compressed with bzip2 and also on uncompressed files.

```
$ bzless trace.bz2


$ bzmore trace.bz2


0: ERR: Wed Sep 22 09:59:42 2010:
gs(11153/47752677794640): [chk_sqlcode.scp:92]: Database:
ORA-01653: unable to extend table OPC_OP.OP

C_HIST_MESSAGES (OpC50-15)

0: ERR: Wed Sep 22 09:59:47 2010:
gs(11153/47752677794640): [chk_sqlcode.scp:92]: Database:
ORA-01653: unable to extend table OPC_OP.OP

C_HIST_MESSAGES (OpC50-15)

Retry. (OpC51-22)

Database: ORA-01653: unable to extend table
OPC_OP.OPC_HIST_MESSAGES by 64 in tablespace OPC_6

(OpC50-15)

.

.

--More--
```

Compare bzip2 files using bzcmp

You can compare two bzip2 compressed file using bzcmp command. It internally uses cmp command to compare the content of the compressed contents. Here you can see the output of comparison of the two normal files and compressed files.

```
$ cmp System.txt.001 System.txt.002
System.txt.001 System.txt.002 differ: byte 20, line 2


$ bzcmp System.txt.001.bz2 System.txt.002.bz2
- /tmp/bzdiff.csgqG32029 differ: byte 20, line 2
```

Find the difference between two bzip2 files using bzdiff

In Linux, diff command will compare two files and give you the lowdown on just how different they are. If you give bz2 files to diff command, it will not be in a position to explain the difference.

For bzip2 compressed files, bzdiff command gives the differences of two bzip2 compressed files as shown below.

```
$ bzdiff System.txt.001.bz2 System.txt.002.bz2
2c2
< 0: ERR: Mon Sep 27 12:19:34 2010: gs(11153/1105824064):
[chk_sqlcode.scp:92]: Database: ORA-01654: unable to
extend index OPC_OP.OPCX
_ANNO_NUM by 64 in tablespace OPC_INDEX1
---
> 0: ERR: Wed Sep 22 09:59:42 2010:
gs(11153/47752677794640): [chk_sqlcode.scp:92]: Database:
ORA-01653: unable to extend table OPC_OP.
OPC_HIST_MESSAGES by 64 in tablespace OPC_6
4,5c4
< Retry. (OpC51-22)
< Database: ORA-01654: unable to extend index
OPC_OP.OPCX_ANNO_NUM by 64 in tablespace OPC_INDEX1
---
> 0: ERR: Wed Sep 22 09:59:47 2010:
gs(11153/47752677794640): [chk_sqlcode.scp:92]: Database:
ORA-01653: unable to extend table OPC_OP.
OPC_HIST_MESSAGES by 64 in tablespace OPC_6
```

Any Questions?

Discuss it here: bzip2, bzgrep, bzcmp, bzdiff, bzcat, bzless, bzmore examples

Hack 51. Cpio Examples

cpio command is used to process archive files (for example, *.cpio or *.tar files).

cpio stands for “copy in, copy out”.

cpio performs the following three operations.

1. Copying files to an archive
2. Extracting files from an archive
3. Passing files to another directory tree

cpio takes the list of files from the standard input while creating an archive, and sends the output to the standard output.

Create *.cpio Archive File

You can create a *.cpio archive that contains files and directories using cpio -ov

```
$ cd objects

$ ls
file1.o file2.o file3.o

$ ls | cpio -ov > /tmp/object.cpio
```

As seen above, the ls command passes the three object filenames to cpio command and cpio generates the object.cpio archive.

Extract *.cpio Archive File

cpio extract: To extract a given *.cpio file, use cpio -iv as shown below.

```
$ mkdir output

$ cd output

$ cpio -idv < /tmp/object.cpio
```

Create *.cpio Archive with Selected Files

The following example creates a *.cpio archive only with *.c files.

```
$ find . -iname *.c -print | cpio -ov >/tmp/c_files.cpio
```

Create *.tar Archive File using cpio -F

We already know how to use the tar command effectively.

Did you know that you can also use cpio command to create tar files as shown below?

```
$ ls | cpio -ov -H tar -F sample.tar
```

As seen above, instead of redirecting the standard output you can mention the output archive filename with the option -F.

Extract *.tar Archive File using cpio command

You can also extract a tar file using cpio command as shown below.

```
$ cpio -idv -F sample.tar
```

View the content of *.tar Archive File

To view the content of *.tar file, do the following.

```
$ cpio -it -F sample.tar
```

Create a *.cpio Archive with the Original files that a Symbolic Link Points

cpio archive can be created with the original files that a symbolic link is referring to as shown below.

```
$ ls | cpio -oLv >/tmp/test.cpio
```

Preserve the File Modification Time while restoring *.cpio

The modification time of the files can be preserved when we are restoring the cpio archive files as shown below.

```
$ ls | cpio -omv >/tmp/test.cpio
```

Copy Directory Tree from One to Another

cpio allows you to copy one directory contents into another directory without creating an intermediate archive. It reads the file list from the standard input and pass it to the target directory.

The example below copies the files and sub-directories of objects directory into /mnt/out directory.

```
$ mkdir /mnt/out

$ cd objects

$ find . -depth | cpio -pmdv /mnt/out
```

In the above example:

- cpio option -p makes cpio to use pass through mode. Its like piping cpio -o into cpio -i.
- cpio option -d creates leading directories as needed in the target directory.

Any Questions?

Discuss it here: Linux cpio Examples: How to Create and Extract cpio Archives (and tar archives)

Additional cpio Examples:

How to View, Modify and Recreate initrd.img Using cpio Command

Chapter 7: Command Line History

Hack 52. Bash Command Line History Examples

When you are using Linux command line frequently, using the history effectively can be a major productivity boost. In fact, once you have mastered the 15 examples that I've provided here, you'll find using command line more enjoyable and fun.

Search the history using Control+R

I strongly believe that this may be your most frequently used feature of history. When you've already executed a very long command, you can simply search history using a keyword and re-execute the same command without having to type it fully. Press Control+R and type the keyword.

In the following example, I searched for red, which displayed the previous command "cat /etc/redhat-release" in the history that contained the word red.

```
# [Note: Press Ctrl+R from the command prompt, which will
display the reverse-i-search prompt as shown below]


(reverse-i-search)`red': cat /etc/redhat-release
[Note: Press enter when you see your command, which will
execute the command from the history]


# cat /etc/redhat-release
Fedora release 9 (Sulphur)
```

Sometimes you want to edit a command from history before executing it. For e.g. you can search for httpd, which will display service httpd stop from the command history, select this command and change the stop to start and re-execute it again as shown below.

```
 # [Note: Press Ctrl+R from the command prompt, which will
display the reverse-i-search prompt]


(reverse-i-search)`httpd': service httpd stop
[Note: Press either left arrow or right arrow key when you
see your command, which will display the command for you
to edit, before executing it]


# service httpd start
```

Repeat previous command quickly using 4 different methods

Sometime you may end up repeating the previous commands for various reasons. Following are the 4 different ways to repeat the last executed command.

1. Use the up arrow to view the previous command and press enter to execute it.
2. Type !! and press enter from the command line
3. Type !-1 and press enter from the command line.
4. Press Control+P will display the previous command, press enter to execute it

Execute a specific command from history

In the following example, If you want to repeat the command #4, execute !4 as shown below.

```
# history | more
       1  service network restart
       2  exit
       3  id
       4  cat /etc/redhat-release


# !4
cat /etc/redhat-release
```

```
Fedora release 9 (Sulphur)
```

Execute previous command that starts with a specific word

Type ! followed by the starting few letters of the command that you would like to re-execute. In the following example, typing !ps and enter, executed the previous command starting with ps, which is 'ps aux | grep yp'.

```
# !ps
ps aux | grep yp
root 16947  0.0  0.1  36516  1264 ? Sl 13:10   0:00 ypbind
```

Clear all the previous history using option -c

Sometime you may want to clear all the previous history. However you may still want to keep the history moving forward.

```
# history -c
```

Substitute words from history commands

When you are searching through history, you may want to execute a different command but use the same parameter from the command that you've just searched.

In the example below, the !!:$ next to the vi command gets the argument from the previous command to the current command.

```
# ls anaconda-ks.cfg
anaconda-ks.cfg

# vi !!:$
vi anaconda-ks.cfg
```

In the example below, the !^ next to the vi command gets the first argument from the previous command (i.e cp command) to the current command (i.e vi command).

```
# cp anaconda-ks.cfg anaconda-ks.cfg.bak
anaconda-ks.cfg


# vi !^
vi anaconda-ks.cfg
```

Substitute a specific argument for a specific command

In the example below, !cp:2 searches for the previous command in history that starts with cp and takes the second argument of cp and substitutes it for the ls -l command as shown below.

```
# cp ~/longname.txt /really/a/very/long/path/long-
filename.txt


# ls -l !cp:2
ls -l /really/a/very/long/path/long-filename.txt
```

In the example below, !cp:$ searches for the previous command in history that starts with cp and takes the last argument (in this case, which is also the second argument as shown above) of cp and substitutes it for the ls -l command as shown below.

```
# ls -l !cp:$
ls -l /really/a/very/long/path/long-filename.txt
```

Hack 53. History Related Environment Variables

Display TIMESTAMP in history using HISTTIMEFORMAT

Typically when you type history from command line, it displays the command# and the command. For auditing purpose, it may be beneficial to display the timestamp along with the command as shown below.

```
# export HISTTIMEFORMAT='%F %T '

# history | more
      1  2008-08-05 19:02:39 service network restart
      2  2008-08-05 19:02:39 exit
      3  2008-08-05 19:02:39 id
      4  2008-08-05 19:02:39 cat /etc/redhat-release
```

Note: You can also setup the following alias to view the recent history commands.

```
alias h1='history 10'
alias h2='history 20'
alias h3='history 30'
```

Control the total number of lines in the history using HISTSIZE

Append the following two lines to the .bash_profile and relogin to the bash shell again to see the change. In this example, only 450 command will be stored in the bash history.

```
# vi ~/.bash_profile

HISTSIZE=450
HISTFILESIZE=450
```

Change the history file name using HISTFILE

By default, history is stored in ~/.bash_history file. Add the following line to the .bash_profile and relogin to the bash shell, to store the history command in .commandline_warrior file instead of .bash_history file. I'm yet to figure out a practical use for this. I can see this getting used when you want to track commands executed from different terminals using different history file name.

```
# vi ~/.bash_profile
HISTFILE=/root/.commandline_warrior
```

Eliminate the continuous repeated entry from history using HISTCONTROL

In the following example pwd was typed three times, when you do history, you can see all the 3 continuous occurrences of it. To eliminate duplicates, set HISTCONTROL to ignoredups as shown below.

```
# pwd

# pwd

# pwd

# history | tail -4
       44  pwd
       45  pwd
       46  pwd
       47  history | tail -4

[Note: There are three pwd commands in history, after
executing pwd 3 times as shown above]

# export HISTCONTROL=ignoredups

# pwd
```

```
# pwd

# pwd

# history | tail -3
       56  export HISTCONTROL=ignoredups
       57  pwd
       58  history | tail -4
```

[**Note**: There is only one pwd command in the history, even after executing pwd 3 times as shown above]

Erase duplicates across the whole history using HISTCONTROL

The ignoredups shown above removes duplicates only if they are consecutive commands. To eliminate duplicates across the whole history, set the HISTCONTROL to erasedups as shown below.

```
# export HISTCONTROL=erasedups

# pwd

# service httpd stop

# history | tail -3
       38  pwd
       39  service httpd stop
       40  history | tail -3

# ls -ltr

# service httpd stop
```

```
# history | tail -6
       35  export HISTCONTROL=erasedups
       36  pwd
       37  history | tail -3
       38  ls –ltr
       39  service httpd stop
       40  history | tail -6

[Note: The previous service httpd stop after pwd got
erased]
```

Force history not to remember a particular command using HISTCONTROL

When you execute a command, you can instruct history to ignore the command by setting HISTCONTROL to ignorespace AND typing a space in front of the command as shown below. I can see lot of junior sysadmins getting excited about this, as they can hide a command from the history.

It is good to understand how ignorespace works. But, as a best practice, don't hide purposefully anything from history.

```
# export HISTCONTROL=ignorespace

# ls –ltr

# pwd

#  service httpd stop
[Note:  There is a space at the beginning of service, to
ignore this command from history]

# history | tail -3
```

```
       67  ls –ltr
       68  pwd
       69  history | tail -3
```

Disable the usage of history using HISTSIZE

If you want to disable history all together and don't want bash shell to remember the commands you've typed, set the HISTSIZE to 0 as shown below.

```
# export HISTSIZE=0

# history
# [Note:  History did not display anything]
```

Ignore specific commands from the history using HISTIGNORE

Sometimes you may not want to clutter your history with basic commands such as pwd and ls. Use HISTIGNORE to specify all the commands that you want to ignore from the history.

Please note that adding ls to the HISTIGNORE ignores only ls and not ls -l. So, you have to provide the exact command that you would like to ignore from the history.

```
# export HISTIGNORE="pwd:ls:ls –ltr:"

# pwd
# ls
# ls -ltr
# service httpd stop

# history | tail -3
       79  export HISTIGNORE="pwd:ls:ls -ltr:"
       80  service httpd stop
```

```
       81  history
[Note: History did not display pwd and ls]
```

Any Questions?

Discuss it here: 15 Examples To Master Linux Command Line History

Hack 54. History Expansion Examples

Using history expansion you can pick a specific command from the history, execute it as it is, or modify it and execute it based on your needs. The ! starts the history expansion.

- **!!** Repeats the previous command
- **!10** Repeat the 10th command from the history
- **!-2** Repeat the 2nd command (from the last) from the history
- **!string** Repeat the command that starts with “string” from the history
- **!?string** Repeat the command that contains the word “string” from the history
- **^str1^str2^** Substitute str1 in the previous command with str2 and execute it
- **!!:$** Gets the last argument from the previous command.
- **!string:n** Gets the nth argument from the command that starts with “string” from the history.

!?string Example

Let us assume that you've executed the following command at some point and it is somewhere in the history.

```
$ /usr/local/apache2/bin/apachectl restart
```

Later when you want to execute the same command, if you try the following it will fail because it is looking for a line that starts with "apache".

```
$ !apache
-bash: !apache: event not found
```

However if you do the following, it will look for any command that contains the string "apache" and execute it as shown below.

```
$ !?apache
/usr/local/apache2/bin/apachectl restart
```

^str1^str2^ Example

Sometimes you might check whether a file exists using a quick ls command as shown below.

```
$ ls /etc/sysconfig/network
```

Once you verify that the file exists, to view the content of the file using vi, you don't need to type the whole file name again. Instead do the following, which will replace the word 'ls' in the previous command with the word 'vi' and execute the command.

```
$ ^ls^vi
vi /etc/sysconfig/network
```

!!:$ Example

In this example, the following command takes a copy of the /etc/passwd file to the home directory as passwd.bak.

```
$ cp /etc/passwd /home/ramesh/passwd.bak
```

Once you create the backup of the file, if you want to open the backup file, you don't need to type the whole backup file name again. Instead,

you can use the last argument of the previous command along with 'vi' command as shown below.

```
$ vi !!:$
vi /home/ramesh/passwd.bak
```

Please note that “!:$” is exactly same as “!!:$”. So, the above example can also be executed as shown below.

```
$ vi !:$
vi /home/ramesh/passwd.bak
```

!string:n Example

When you execute a command that has multiple arguments (as shown in the tar command example below), you can extract only a specific argument from it for later use.

```
$ tar cvfz ~/sysconfig.tar.gz /etc/sysconfig/*
```

Now if you want to ls the newly created tar.gz file, you can do the following, which will take the 2nd argument of the previous tar command.

```
$ ls -l !tar:2
ls -l ~/sysconfig.tar
```

Additional Bash History Expansion Examples:

15 Linux Bash History Expansion Examples You Should Know

Chapter 8: System Administration Tasks

Hack 55. Partition Using fdisk

After you've installed brand new disks on your server, you have to use tools like fdisk to partition it accordingly.

Following are the 5 typical actions (commands) that you can execute inside fdisk.

- n – New Partition creation
- d – Delete an existing partition
- p - Print Partition Table
- w – Write the changes to the partition table. i.e save.
- q – Quit the fdisk utility

Create a partition

In the following example, I created a /dev/sda1 primary partition.

```
# fdisk /dev/sda
Command (m for help): p

Disk /dev/sda: 287.0 GB, 287005343744 bytes
255 heads, 63 sectors/track, 34893 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

Device Boot  Start   End      Blocks   Id  System

Command (m for help): n
Command action
   e   extended
```

```
   p   primary partition (1-4)
p

Partition number (1-4): 1
First cylinder (1-34893, default 1):
Using default value 1
Last cylinder or +size or +sizeM or +sizeK (1-34893,
default 34893):
Using default value 34893

Command (m for help): w
The partition table has been altered!

Calling ioctl() to re-read partition table.
Syncing disks.
```

Verify that the partition got created successfully

```
# fdisk /dev/sda
Command (m for help): p

Disk /dev/sda: 287.0 GB, 287005343744 bytes
255 heads, 63 sectors/track, 34893 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

Device Boot  Start         End      Blocks   Id  System
/dev/sda1        1       34893   280277991   83  Linux

Command (m for help): q
```

Additional fdisk Examples:

7 Linux fdisk Command Examples to Manage Hard Disk Partition

Hack 56. Format a Partition Using mke2fsk

After partitioning the disks, it is still not ready for usage, as we need to format the disk. At this stage, if you try to view the disk information, it will give the following error message indicating that no valid superblock is present.

```
# tune2fs -l /dev/sda1


tune2fs 1.35 (28-Feb-2004)
tune2fs: Bad magic number in super-block while trying to
open /dev/sda1


Couldn't find valid filesystem superblock.
```

To format the disk, use mke2fs as shown below.

```
# mke2fs /dev/sda1
```

You can also pass the following optional parameter to the mke2fs.

- -m 0 : reserved-blocks-percentage – This indicates the percentage of the filesystem blocks reserved for the root user. Default is 5%. In the following example, it is set to 0.
- -b 4096 : block-size specified in bytes. Valid values are 1024, 2048 and 4096 bytes per block.

```
# mke2fs -m 0 -b 4096 /dev/sda1
mke2fs 1.35 (28-Feb-2004)
```

```
Filesystem label=
OS type: Linux
Block size=4096 (log=2)
Fragment size=4096 (log=2)
205344 inodes, 70069497 blocks
0 blocks (0.00%) reserved for the super user
First data block=0
Maximum filesystem blocks=71303168
2139 block groups
32768 blocks per group, 32768 fragments per group
96 inodes per group
Superblock backups stored on blocks:
32768, 98304, 163840, 229376, 294912, 819200, 884736,
1605632, 2654208, 4096000, 7962624, 11239424, 20480000,
23887872


Writing inode tables: done
Writing superblocks and filesystem accounting information:
done


This filesystem will be automatically checked every 32
mounts or 180 days, whichever comes first.  Use tune2fs -c
or -i to override.
```

The above command will create an ext2 filesystem. To create an ext3 file system do the following:

```
# mkfs.ext3 /dev/sda1


# mke2fs –j /dev/sda1
```

Hack 57. Mount a Partition

After creating a partition and formatting, you can mount it to a mount point.

First create a directory where the partition should be mounted.

```
# mkdir /home/database
```

Mount the file system.

```
# mount /dev/sda1 /home/database
```

To automatically mount the filesystem after the reboot, add the following entry to the /etc/fstab

```
/dev/sda1 /home/database ext3 defaults 0 2
```

Hack 58. Fine Tune a Partition Using tune2fs

Use the tune2fs –l /dev/sda1 to view the filesystem information as shown below.

```
# tune2fs -l /dev/sda1
tune2fs 1.35 (28-Feb-2004)
Filesystem volume name:   /home/database
Last mounted on:          <not available>
Filesystem UUID:          f1234556-e123-1234-abcd-
bbbbaaaaae11
Filesystem magic number:  0xEF44
Filesystem revision #:    1 (dynamic)
Filesystem features:      resize_inode filetype
sparse_super
Default mount options:    (none)
Filesystem state:         not clean
```

```
Errors behavior:          Continue
Filesystem OS type:       Linux
Inode count:              1094912
Block count:              140138994
Reserved block count:     0
Free blocks:              16848481
Free inodes:              1014969
First block:              0
Block size:               2048
Fragment size:            2048
Reserved GDT blocks:      512
Blocks per group:         16384
Fragments per group:      16384
Inodes per group:         128
Inode blocks per group:   8
Filesystem created:       Tue Jul  1 00:06:03 2008
Last mount time:          Thu Aug 21 05:58:25 2008
Last write time:          Fri Jan  2 15:40:36 2009
Mount count:              2
Maximum mount count:      20
Last checked:             Tue Jul  1 00:06:03 2008
Check interval:           15552000 (6 months)
Next check after:         Sat Dec 27 23:06:03 2008
Reserved blocks uid:      0 (user root)
Reserved blocks gid:      0 (group root)
First inode:              11
Inode size:               128
Default directory hash:   tea
Directory Hash Seed:      12345829-1236-4123-9aaa-
ccccc123292b
```

You can also use the tune2fs to tune the ex2/ext3 filesystem parameter. For example, if you want to change the Filesystem volume name, you can do it as shown below.

```
# tune2fs -l /dev/sda1 | grep volume
Filesystem volume name:   /home/database

# tune2fs -L database-home /dev/emcpowera1
tune2fs 1.35 (28-Feb-2004)

# tune2fs -l /dev/sda1 | grep volume
Filesystem volume name:   database-home
```

Hack 59. Create a Swap File System

Create a file for swap usage as shown below.

```
# dd if=/dev/zero of=/home/swap-fs bs=1M count=512
512+0 records in
512+0 records out

# ls -l /home/swap-fs
-rw-r--r--  1 root root 536870912 Jan  2 23:13 /home/swap-
fs
```

Use mkswap to setup a Linux swap area in the /home/swap-fs file that was created above.

```
# mkswap /home/swap-fs
```

Setting up swapspace version 1, size = 536866 kB

Once the file is created and has been setup for Linux swap area, it is time to enable the swap using swapon as shown below.

```
# swapon /home/swap-fs
```

Add the following line to /etc/fstab and reboot the system for the swap to take into effect.

```
/home/swap-fs swap swap defaults 0 0
```

Additional Swap Examples:

2 Ways to Add Swap Space Using dd, mkswap and swapon

Hack 60. Create a New User

Add a new user – Basic method

Specify only the user name.

```
# useradd jsmith
```

Add a new user with additional Parameter

You can also specify the following parameter to the useradd

- -c : Description about the user.
- -e : expiry date of the user in mm/dd/yy format

```
# adduser -c "John Smith - Oracle Developer" -e 12/31/09
jsmith
```

Verify that the user got added successfully.

```
# grep jsmith /etc/passwd
jsmith:x:510:510:John Smith - Oracle
Developer:/home/jsmith:/bin/bash
```

Change the user password

```
# passwd jsmith
Changing password for user jsmith.
New UNIX password:
BAD PASSWORD: it is based on a dictionary word
Retype new UNIX password:
passwd: all authentication tokens updated successfully.
```

How to identify the default values used by useradd?

Following are the default values that will be used when an user is created.

```
# useradd –D
GROUP=100
HOME=/home
INACTIVE=-1
EXPIRE=
SHELL=/bin/bash
SKEL=/etc/skel
```

Additional User Creation Examples:

The Ultimate Guide to Create Users in Linux / Unix

The Ultimate Guide for Creating Strong Passwords

Hack 61. Create a New Group

Create a new developer group

```
# groupadd developers
```

Validate that the group was created successfully.

```
# grep developer /etc/group
developers:x:511:
```

Add an user to an existing group

You cannot use useradd to modify an existing user, as you'll get the following error message.

```
# useradd -G developers jsmith
useradd: user jsmith exists

# usermod -g developers jsmith
```

Validate the users group was modified successfully

```
# grep jsmith /etc/passwd
jsmith:x:510:511:Oracle Developer:/home/jsmith:/bin/bash

# id jsmith
uid=510(jsmith) gid=511(developers) groups=511(developers)

# grep jsmith /etc/group
jsmith:x:510:
developers:x:511:jsmith
```

Hack 62. Setup SSH Passwordless Login in OpenSSH

You can login to a remote Linux server without entering password in 3 simple steps using ssky-keygen and ssh-copy-id as explained in this example.

ssh-keygen creates the public and private keys. ssh-copy-id copies the local-host's public key to the remote-host's authorized_keys file. ssh-copy-id also assigns proper permission to the remote-host's home, ~/.ssh, and ~/.ssh/authorized_keys.

Step 1: Create public and private keys using ssh-key-gen on local-host

```
jsmith@local-host$ ssh-keygen

Generating public/private rsa key pair.

Enter file in which to save the key
(/home/jsmith/.ssh/id_rsa):[Enter key]

Enter passphrase (empty for no passphrase): [Press enter
key]

Enter same passphrase again: [Pess enter key]

Your identification has been saved in
/home/jsmith/.ssh/id_rsa.

Your public key has been saved in
/home/jsmith/.ssh/id_rsa.pub.

The key fingerprint is:

33:b3:fe:af:95:95:18:11:31:d5:de:96:2f:f2:35:f9
jsmith@local-host
```

Step 2: Copy the public key to remote-host using ssh-copy-id

```
jsmith@local-host$ ssh-copy-id -i ~/.ssh/id_rsa.pub
remote-host

jsmith@remote-host's password:
```

```
Now try logging into the machine, with “ssh ‘remote-
host’”, and check in:
.ssh/authorized_keys to make sure we haven’t added extra
keys that you weren’t expecting.
```

Note: ssh-copy-id appends the keys to the remote-host’s .ssh/authorized_key.

Step 3: Login to remote-host without entering the password

```
jsmith@local-host$ ssh remote-host
 Last login: Sun Nov 16 17:22:33 2008 from 192.168.1.2

[Note: SSH did not ask for password.]

jsmith@remote-host$ [Note: You are on remote-host here]
```

Any Questions?

Discuss it here: 3 Steps to Perform SSH Login Without Password Using ssh-keygen & ssh-copy-id

Hack 63. Use ssh-copy-id Along With ssh-agent

Using ssh-copy-id along with the ssh-add/ssh-agent

When no value is passed for the option -i and If ~/.ssh/identity.pub is not available, ssh-copy-id will display the following error message.

```
jsmith@local-host$ ssh-copy-id -i remote-host
```

```
/usr/bin/ssh-copy-id: ERROR: No identities found
```

If you have loaded keys to the ssh-agent using the ssh-add, then ssh-copy-id will get the keys from the ssh-agent to copy to the remote-host. i.e, it copies the keys provided by ssh-add -L command to the remote-host, when you don't pass option -i to the ssh-copy-id.

```
jsmith@local-host$ ssh-agent $SHELL


jsmith@local-host$ ssh-add -L
The agent has no identities.


jsmith@local-host$ ssh-add
Identity added: /home/jsmith/.ssh/id_rsa
(/home/jsmith/.ssh/id_rsa)


jsmith@local-host$ ssh-add -L
ssh-rsa
AAAAB3NzaC1yc2EAAAABIwAAAQEAsJIEILxftj8aSxMa3d8t6JvM79D
aHrtPhTYpq7kIEMUNzApnyxsHpH1tQ/Ow==
/home/jsmith/.ssh/id_rsa


jsmith@local-host$ ssh-copy-id -i remote-host
jsmith@remote-host's password:
Now try logging into the machine, with "ssh 'remote-
host'", and check in: .ssh/authorized_keys to make sure we
haven't added extra keys that you weren't expecting.
[Note: This has added the key displayed by ssh-add -L]
```

Three Minor Annoyances of ssh-copy-id

Following are few minor annoyances of the ssh-copy-id.

1. **Default public key:** ssh-copy-id uses ~/.ssh/identity.pub as the default public key file (i.e when no value is passed to option -i). Instead, I wish it uses id_dsa.pub, or id_rsa.pub, or identity.pub as default keys. i.e If any one of them exist, it should copy that to

the remote-host. If two or three of them exist, it should copy identity.pub as default.

2. **The agent has no identities:** When the ssh-agent is running and the ssh-add -L returns "The agent has no identities" (i.e no keys are added to the ssh-agent), the ssh-copy-id will still copy the message "The agent has no identities" to the remote-host's authorized_keys entry.
3. **Duplicate entry in authorized_keys:** I wish ssh-copy-id validates duplicate entry on the remote-host's authorized_keys. If you execute ssh-copy-id multiple times on the local-host, it will keep appending the same key on the remote-host's authorized_keys file without checking for duplicates. Even with duplicate entries everything works as expected. But, I would like to have my authorized_keys file clutter free.

Hack 64. Crontab Examples

Using cron you can execute a shell-script or Linux commands at a specific time and date. For example a sysadmin can schedule a backup job that can run every day.

How to add a job to the cron?

```
# crontab –e
0 5 * * * /root/bin/backup.sh
```

This will execute /root/bin/backup.sh at 5 a.m every day.

Description of Cron fields

Following is the format of the crontab file.

```
{minute} {hour} {day-of-month} {month} {day-of-week}
{full-path-to-shell-script}
```

- minute: Allowed range 0 – 59
- hour: Allowed range 0 – 23
- day-of-month: Allowed range 0 – 31

- month: Allowed range 1 – 12. 1 = January. 12 = December.
- Day-of-week: Allowed range 0 – 7. Sunday is either 0 or 7.

Crontab examples

1. Run at 12:01 a.m. 1 minute after midnight everyday. This is a good time to run backup when the system is not under load.

```
1 0 * * * /root/bin/backup.sh
```

2. Run backup every weekday (Mon – Fri) at 11:59 p.m.

```
 59 11 * * 1,2,3,4,5 /root/bin/backup.sh
```

Following will also do the same.

```
59 11 * * 1-5 /root/bin/backup.sh
```

3. Execute the command every 5 minutes.

```
*/5 * * * * /root/bin/check-status.sh
```

4. Execute at 1:10 p.m on 1st of every month

```
10 13 1 * * /root/bin/full-backup.sh
```

5. Execute 11 p.m on weekdays.

```
0 23 * * 1-5 /root/bin/incremental-backup.sh
```

Crontab Options

The following are the available options with crontab:

- crontab –e : Edit the crontab file. This will create a crontab, if it doesn’t exist

- crontab –l : Display the crontab file.
- crontab -r : Remove the crontab file.
- crontab -ir : This will prompt user before deleting a crontab.

Additional Cron Examples:

Linux Crontab: 15 Awesome Cron Job Examples

How to Run Cron Every 5 Minutes, Seconds, Hours, Days, Months

Cron Vs Anacron: How to Setup Anacron on Linux (With an Example)

Hack 65. Safe Reboot Of Linux Using Magic SysRq Key

The magic SysRq key is a key combination in the Linux kernel which allows the user to perform various low level commands regardless of the system's state.

It is often used to recover from freezes, or to reboot a computer without corrupting the filesystem. The key combination consists of Alt+SysRq+commandkey. In many systems the SysRq key is the printscreen key.

First, you need to enable the SysRq key, as shown below.

```
echo "1" > /proc/sys/kernel/sysrq
```

List of SysRq Command Keys

The following are the command keys available for Alt+SysRq+commandkey:

- ‘k’ – Kills all the process running on the current virtual console.
- ’s’ – This will attempt to sync all the mounted file system.
- ‘b’ - Immediately reboot the system, without unmounting partitions or syncing.
- ‘e’ – Sends SIGTERM to all process except init.
- ‘m’ – Output current memory information to the console.
- ‘i’ - Send the SIGKILL signal to all processes except init
- ‘r’ - Switch the keyboard from raw mode (the mode used by programs such as X11), to XLATE mode.
- ’s’ – sync all mounted file system.
- ‘t’ - Output a list of current tasks and their information to the console.
- ‘u’ - Remount all mounted filesystems in readonly mode.
- ‘o’ – Shutdown the system immediately.
- ‘p’ – Print the current registers and flags to the console.
- ‘0-9′ - Sets the console log level, controlling which kernel messages will be printed to your console.
- ‘f’ - Will call oom_kill to kill process which takes more memory.
- ‘h’ – Used to display the help. But any other keys than the above listed will print help.

We can also do this by echoing the keys to the /proc/sysrq-trigger file. For example, to re-boot a system you can perform the following.

```
echo "b" > /proc/sysrq-trigger
```

Perform a Safe reboot of Linux using Magic SysRq Key

To perform a safe reboot of a Linux computer which hangs up, do the following. This will avoid the fsck during the next re-booting. i.e Press Alt+SysRq+letter highlighted below.

- un**R**aw (take control of keyboard back from X11)
- t**E**rminate (send SIGTERM to all processes, allowing them to terminate gracefully),

- **k**lll (send SIGILL to all processes, forcing them to terminate immediately),
- **S**ync (flush data to disk),
- **U**nmount (remount all filesystems read-only),
- re**B**oot.

Any Questions?

Discuss it here: Safe Reboot Of Linux Using Magic SysRq Key

Hack 66. Linux Parted Command Examples

Parted is a GNU utility, which is used to manipulate the hard disk partitions.

Using parted, you can add, delete, and edit partitions and the file systems located on those partitions. You can also clone partitions.

This hack explains 9 practical parted command examples.

Warning: Parted utility manipulates the hard disk partition table and saves the changes immediately. So, don't delete, modify, add, or do anything to your partition, if you don't know what you are doing. You will lose your data! There is no undo button for your rescue!

Select the hard disk to be parted

When you execute parted command without any argument, by default it selects the first hard disk drive that is available on your system.

In the following example, it picked /dev/sda automatically as it is the first hard drive in this system.

```
# parted
GNU Parted 2.3
Using /dev/sda
Welcome to GNU Parted! Type 'help' to view a list of
commands.
(parted)
To choose a different hard disk, use the select command as
shown below.


(parted) select /dev/sdb
```

It will throw the following error message when it doesn't find the given hard disk name.

```
Error: Error opening /dev/sdb: No medium found
Retry/Cancel? y
```

Display all Partitions Using print

Using the print command, you can view all the available partitions in the selected hard disk. The print command also displays hard disk properties such as model, size, sector size and partition table as shown below.

```
(parted) print
Model: ATA WDC WD5000BPVT-7 (scsi)
Disk /dev/sda: 500GB
Sector size (logical/physical): 512B/4096B
Partition Table: msdos

Number  Start   End     Size    Type      Filesystem Flags
 1      1049kB  106MB   105MB   primary   fat16      diag
 2      106MB   15.8GB  15.7GB  primary   ntfs       boot
 3      15.8GB  266GB   251GB   primary   ntfs
 4      266GB   500GB   234GB   extended
```

```
5      266GB   269GB   2682MB  logical   ext4
7      269GB   270GB   524MB   logical   ext4
8      270GB   366GB   96.8GB  logical   lvm
6      366GB   370GB   3999MB  logical   linux-swap(v1)
9      370GB   500GB   130GB   logical   ext4
```

Create Primary Partition in Selected HDD Using mkpart

mkpart command is used to create either primary or logical partition with the START and END disk locations. The below example creates partition with size around 15GB. The START and END points passed to the mkpart command are in the units of MBs.

```
(parted) mkpart primary 106 16179
```

You can also enable boot option on a partition as shown below. Linux reserves 1-4 or 1-3 partition number for primary partition and the extended partition starts from number 5.

```
(parted) set 1 boot on
```

Create Logical Partition in Selected HDD Using mkpart

Use mkpart command to create a new partition of a specific size. This will create the partition of a specific type such as primary, logical or extended without creating the file system.

Before creating the partition, execute a print command to view the current layout.

```
(parted) print
Model: ATA WDC WD5000BPVT-7 (scsi)
Disk /dev/sda: 500GB
Sector size (logical/physical): 512B/4096B
Partition Table: msdos
```

```
Number  Start   End     Size    Type      Filesystem Flags
 1      1049kB  106MB   105MB   primary   fat16      diag
 2      106MB   15.8GB  15.7GB  primary   ntfs       boot
 3      15.8GB  266GB   251GB   primary   ntfs
 4      266GB   500GB   234GB   extended
 5      266GB   316GB   50.0GB  logical   ext4
 6      316GB   324GB   7999MB  logical   linux-swap(v1)
 7      324GB   344GB   20.0GB  logical   ext4
 8      344GB   364GB   20.0GB  logical   ext2
```

Use mkpart to create a new logical partition with 127GB size as shown below.

```
(parted) mkpart logical 372737 500000
```

Execute the print command to view the new layout as shown below.

```
(parted) print
Model: ATA WDC WD5000BPVT-7 (scsi)
Disk /dev/sda: 500GB
Sector size (logical/physical): 512B/4096B
Partition Table: msdos

Number  Start   End     Size    Type      Filesystem Flags
 1      1049kB  106MB   105MB   primary   fat16      diag
 2      106MB   15.8GB  15.7GB  primary   ntfs       boot
 3      15.8GB  266GB   251GB   primary   ntfs
 4      266GB   500GB   234GB   extended
 5      266GB   316GB   50.0GB  logical   ext4
 6      316GB   324GB   7999MB  logical   linux-swap(v1)
 7      324GB   344GB   20.0GB  logical   ext4
 8      344GB   364GB   20.0GB  logical   ext2
 9      373GB   500GB   127GB   logical
```

```
(parted)
```

Create a File System on Partition Using mkfs

If you use fdisk command to partition your hard disk, you need to exit the fdisk utility, and use the mkfs external program to create a file system on the partition.

However using parted utility, you can also create filesystem. Use the parted's mkfs command to create a file system on a partition. You should be careful while doing this, as all the existing data in the partition will be lost during the file system creation. The supported filesystems in parted are ext2, mips, fat16, fat32, linux-swap, reiserfs (if libreiserfs is installed).

Let us change the file system of partition number 8 (that is shown in the print output below) from ext4 to ext2 file system.

```
(parted) print
Model: ATA WDC WD5000BPVT-7 (scsi)
Disk /dev/sda: 500GB
Sector size (logical/physical): 512B/4096B
Partition Table: msdos

Number  Start   End     Size    Type      Filesystem Flags
 1      1049kB  106MB   105MB   primary   fat16      diag
 2      106MB   15.8GB  15.7GB  primary   ntfs       boot
 3      15.8GB  266GB   251GB   primary   ntfs
 4      266GB   500GB   234GB   extended
 5      266GB   316GB   50.0GB  logical   ext4
 6      316GB   324GB   7999MB  logical   linux-swap(v1)
 7      324GB   344GB   20.0GB  logical   ext4
 8      344GB   364GB   20.0GB  logical   ext4
```

```
 9      364GB   500GB   136GB   logical   ext4
```

As shown below, use the mkfs command to change the file system type of partition number 8. mkfs command will prompt you for partition number and file system type.

```
(parted) mkfs
Warning: The existing file system will be destroyed and
all data on the partition will be lost. Do you want to
continue?
Yes/No? y
Partition number? 8
File system type?  [ext2]? ext2
```

Execute the print command again, to verify that the file system type for partition number 8 was changed to ex2.

```
(parted) print
Model: ATA WDC WD5000BPVT-7 (scsi)
Disk /dev/sda: 500GB
Sector size (logical/physical): 512B/4096B
Partition Table: msdos

Number  Start   End     Size    Type      Filesystem Flags
 1      1049kB  106MB   105MB   primary   fat16      diag
 2      106MB   15.8GB  15.7GB  primary   ntfs       boot
 3      15.8GB  266GB   251GB   primary   ntfs
 4      266GB   500GB   234GB   extended
 5      266GB   316GB   50.0GB  logical   ext4
 6      316GB   324GB   7999MB  logical   linux-swap(v1)
 7      324GB   344GB   20.0GB  logical   ext4
 8      344GB   364GB   20.0GB  logical   ext2
 9      364GB   500GB   136GB   logical   ext4
```

```
(parted)
```

Create Partition and Filesystem together Using mkpartfs

Using mkpartfs parted command, you can also create a partitions with a specific filesystem. This is similar to mkpart, but with the additional feature of creating file system on a partition.

Before mkpartfs following is the layout of the partitions.

```
(parted) print
Model: ATA WDC WD5000BPVT-7 (scsi)
Disk /dev/sda: 500GB
Sector size (logical/physical): 512B/4096B
Partition Table: msdos

Number  Start   End     Size    Type      Filesystem Flags
 1      1049kB  106MB   105MB   primary   fat16      diag
 2      106MB   15.8GB  15.7GB  primary   ntfs       boot
 3      15.8GB  266GB   251GB   primary   ntfs
 4      266GB   500GB   234GB   extended
 5      266GB   316GB   50.0GB  logical   ext4
 6      316GB   324GB   7999MB  logical   linux-swap(v1)
 7      324GB   344GB   20.0GB  logical   ext4
 8      344GB   364GB   20.0GB  logical
```

In the following example, mkpartfs will create a new fat32 partition of size 127GB.

```
(parted) mkpartfs logical fat32 372737 500000
```

As you see below, the partition number 9 is successfully created.

```
(parted) print
Model: ATA WDC WD5000BPVT-7 (scsi)
Disk /dev/sda: 500GB
Sector size (logical/physical): 512B/4096B
Partition Table: msdos

Number  Start   End     Size    Type      Filesystem Flags
 1      1049kB  106MB   105MB   primary   fat16      diag
 2      106MB   15.8GB  15.7GB  primary   ntfs       boot
 3      15.8GB  266GB   251GB   primary   ntfs
 4      266GB   500GB   234GB   extended
 5      266GB   316GB   50.0GB  logical   ext4
 6      316GB   324GB   7999MB  logical   linux-swap(v1)
 7      324GB   344GB   20.0GB  logical   ext4
 8      344GB   364GB   20.0GB  logical
 9      373GB   500GB   127GB   logical   fat32      lba

(parted)
```

Resize Partition from One Size to Another Using resize

Using resize parted command, you can increase or decrease the partition size of a partition as shown in the example below.

```
(parted) resize 9
Start?  [373GB]? 373GB
End?  [500GB]? 450GB
```

As shown above, parted command will always warn whenever you are attempting to do something dangerous (i.e : rm, resize, mkfs).

The size of partition 9 is actually reduced from 127GB to 77GB. Verify that the partition is resized properly using the print command as shown below.

```
(parted) print
Model: ATA WDC WD5000BPVT-7 (scsi)
Disk /dev/sda: 500GB
Sector size (logical/physical): 512B/4096B
Partition Table: msdos

Number  Start   End     Size    Type      Filesystem Flags
 1      1049kB  106MB   105MB   primary   fat16      diag
 2      106MB   15.8GB  15.7GB  primary   ntfs       boot
 3      15.8GB  266GB   251GB   primary   ntfs
 4      266GB   500GB   234GB   extended
 5      266GB   316GB   50.0GB  logical   ext4
 6      316GB   324GB   7999MB  logical   linux-swap(v1)
 7      324GB   344GB   20.0GB  logical   ext4
 8      344GB   364GB   20.0GB  logical
 9      373GB   450GB   77.3GB  logical   fat32      lba
```

Parted allows you to type unambiguous abbreviation for commands like “p” for print, “sel” for select,etc.

Copy Data from One Partition to Another Using cp

The entire data from one partition can be copied to another partition using the cp command. You should also remember that the content of the destination will be deleted before copy starts. Make sure that the destination partition has enough size to hold the data from the source partition.

Using the “p” command (print) to display the current partition layout.

```
(parted) p
Model: ATA WDC WD5000BPVT-7 (scsi)
Disk /dev/sda: 500GB
Sector size (logical/physical): 512B/4096B
```

```
Partition Table: msdos

Number  Start   End     Size    Type      Filesystem Flags
 1      1049kB  106MB   105MB   primary   fat16      diag
 2      106MB   15.8GB  15.7GB  primary   ntfs       boot
 3      15.8GB  266GB   251GB   primary   ntfs
 4      266GB   500GB   234GB   extended
 5      266GB   316GB   50.0GB  logical   ext4
 6      316GB   324GB   7999MB  logical   linux-swap(v1)
 7      324GB   344GB   20.0GB  logical   ext4
 8      344GB   364GB   20.0GB  logical   ext2
 9      373GB   450GB   77.3GB  logical   fat32      lba
10      461GB   500GB   39.2GB  logical   ext2
```

It is recommended to unmount both source and destination partition before doing copy. In this example we are going to copy the content from partition 8 to partition 10.

The following shows the content of the corresponding partitions before copy.

```
# mount /dev/sda8 /mnt
# cd /mnt
# ls -l
total 52
-rw-r--r-- 1 root root     0 2011-09-26 22:52 part8
-rw-r--r-- 1 root root    20 2011-09-26 22:52 test.txt

# umount /mnt
# mount /dev/sda10 /mnt
# cd /mnt
# ls -l
total 48
```

```
-rw-r--r-- 1 root root     0 2011-09-26 22:52 part10
```

Use the parted cp command to copy partition 8 to partition 10 as shown below.

```
(parted) cp 8 10
growing file system... 95%      (time left 00:38)
```

The following shows the content of the partition 10 after the copy. As you see below, the content of partition 8 is copied over (overwritten) to the partition 10.

```
 # mount /dev/sda10 /mnt
# cd /mnt
# ls -l
total 52
-rw-r--r-- 1 root root     0 2011-09-26 22:52 part8
-rw-r--r-- 1 root root    20 2011-09-26 22:52 test.txt
```

Note: When you copy across partitions of different filesystem(for example src : ext2 and dst : ext4), the destination partition's file system is actually converted to the file system of source partition (i.e : ext2) .

Remove Partition from a Selected Hard Disk Using rm

To delete an unwanted or unused partition, use the parted rm command and specify the partition number as shown below.

```
(parted) rm
Partition number? 9
(parted)
```

As you see below, the partition number 9 is now deleted.

```
(parted) print
Model: ATA WDC WD5000BPVT-7 (scsi)
Disk /dev/sda: 500GB
Sector size (logical/physical): 512B/4096B
Partition Table: msdos

Number  Start   End     Size    Type      Filesystem Flags
 1      1049kB  106MB   105MB   primary   fat16      diag
 2      106MB   15.8GB  15.7GB  primary   ntfs       boot
 3      15.8GB  266GB   251GB   primary   ntfs
 4      266GB   500GB   23GB   extended
 5      266GB   316GB   50.0GB  logical   ext4
 6      316GB   324GB   7999MB  logical   linux-swap(v1)
 7      324GB   344GB   20.0GB  logical   ext4
 8      344GB   364GB   20.0GB  logical   ext2
```

Any Questions?

Discuss it here: 9 Linux Parted Command Examples – mkpart, mkpartfs, resize partitions

Hack 67. Rsync Command Examples

rsync stands for remote sync.

rsync is used to perform the backup operation in UNIX / Linux.

rsync utility is used to synchronize the files and directories from one location to another in an effective way. Backup location could be on local server or on remote server.

Important features of rsync

- **Speed**: First time, rsync replicates the whole content between the source and destination directories. Next time, rsync transfers only the changed blocks or bytes to the destination location, which makes the transfer really fast.
- **Security**: rsync allows encryption of data using ssh protocol during transfer.
- **Less Bandwidth**: rsync uses compression and decompression of data block by block at the sending and receiving end respectively. So the bandwidth used by rsync will be always less compared to other file transfer protocols.
- **Privileges**: No special privileges are required to install and execute rsync

Syntax :

```
$ rsync options source destination
```

Source and destination could be either local or remote. In case of remote, specify the login name, remote server name and location.

Synchronize Two Directories in a Local Server

To sync two directories in a local computer, use the following rsync -zvr command.

```
$ rsync -zvr /var/opt/installation/inventory/ /root/temp
building file list ... done
sva.xml
svB.xml
.
sent 26385 bytes  received 1098 bytes  54966.00 bytes/sec
total size is 44867  speedup is 1.63
```

In the above rsync example:

- -z is to enable compression
- -v verbose
- -r indicates recursive

Now let us see the timestamp on one of the files that was copied from source to destination. As you see below, rsync didn't preserve timestamps during sync.

```
$ ls -l /var/opt/installation/inventory/sva.xml
/root/temp/sva.xml
-r--r--r-- 1 bin  bin  949 Jun 18  2009
/var/opt/installation/inventory/sva.xml
-r--r--r-- 1 root bin  949 Sep  2  2009 /root/temp/sva.xml
```

Preserve timestamps during Sync using rsync -a

rsync option -a indicates archive mode. -a option does the following,

- Recursive mode
- Preserves symbolic links
- Preserves permissions
- Preserves timestamp
- Preserves owner and group

Now, executing the same command provided in example 1 (But with the rsync option -a) as shown below:

```
$ rsync -azv /var/opt/installation/inventory/ /root/temp/
building file list ... done
./
sva.xml
svB.xml
.
sent 26499 bytes  received 1104 bytes  55206.00 bytes/sec
total size is 44867  speedup is 1.63
```

As you see below, rsync preserved timestamps during sync.

```
$ ls -l /var/opt/installation/inventory/sva.xml
/root/temp/sva.xml
-r--r--r-- 1 root  bin  949 Jun 18  2009
/var/opt/installation/inventory/sva.xml
-r--r--r-- 1 root  bin  949 Jun 18  2009
/root/temp/sva.xml
```

Synchronize Only One File

To copy only one file, specify the file name to rsync command, as shown below.

```
$ rsync -v /var/lib/rpm/Pubkeys /root/temp/
Pubkeys

sent 42 bytes  received 12380 bytes  3549.14 bytes/sec
total size is 12288  speedup is 0.99
```

Synchronize Files From Local to Remote

rsync allows you to synchronize files/directories between the local and remote system.

```
$ rsync -avz /root/temp/
thegeekstuff@192.168.200.10:/home/thegeekstuff/temp/
Password:
building file list ... done
./
rpm/
rpm/Basenames
rpm/Conflictname

sent 15810261 bytes  received 412 bytes  2432411.23
bytes/sec
```

```
total size is 45305958  speedup is 2.87
```

While doing synchronization with the remote server, you need to specify username and ip-address of the remote server. You should also specify the destination directory on the remote server. The format is username@machinename:path

As you see above, it asks for password while doing rsync from local to remote server.

Sometimes you don't want to enter the password while backing up files from local to remote server. For example, If you have a backup shell script, that copies files from local to remote server using rsync, you need the ability to rsync without having to enter the password.

To do that, setup ssh password less login as we explained earlier.

Synchronize Files From Remote to Local

When you want to synchronize files from remote to local, specify remote path in source and local path in target as shown below.

```
$ rsync -avz thegeekstuff@192.168.200.10:/var/lib/rpm
/root/temp
Password:
receiving file list ... done
rpm/
rpm/Basenames
.
sent 406 bytes  received 15810230 bytes  2432405.54
bytes/sec
total size is 45305958  speedup is 2.87
```

Additional RSYNC Examples:

How to Backup Linux? 15 rsync Command Examples

6 rsync Examples to Exclude Multiple Files and Directories using exclude-from

Hack 68. Chkconfig Command Examples

Chkconfig command is used to setup, view, or change services that are configured to start automatically during the system startup.

This hack contains 7 practical examples that explains how to use the chkconfig command.

Check Service Startup status from Shell Script

When you execute chkconfig command only with the service name, it returns true if the service is configured for startup. The following code snippet shows how to check whether a service is configured for startup or not from a shell script.

```
# vi check.sh
chkconfig network && echo "Network service is configured"
chkconfig junk && echo "Junk service is configured"

# ./check.sh
Network service is configured
```

You can also specifically check whether it is configured for a particular run level or not.

```
# vi check1.sh
chkconfig network --level 3 && echo "Network service is
configured for level 3"
```

```
chkconfig network --level 1 && echo "Network service is
configured for level 1"


# ./check1.sh
Network service is configured for level 3
```

View Current Status of Startup Services

The –list option displays all the services with the current startup configuration status.

```
# chkconfig --list
abrtd 0:off  1:off   2:off   3:on    4:off   5:on    6:off
acpid 0:off  1:off   2:off   3:off   4:off   5:off   6:off
atd   0:off  1:off   2:off   3:on    4:on    5:on    6:off
...
```

To view only the services that are configured to be started during system startup, do the following. Please note that this assumes that your system startup level is 3.

```
chkconfig --list | grep 3:on
```

To view the startup configuration of a particular service, grep the output of ‘chkconfig –list’ for that service.

```
chkconfig --list | grep network
```

Add a new Service to the Startup

Use –add option to add a specific service to the list of services that will be started during system reboot.

The following example shows how to add a new service (for example, iptables) to the list of services that needs to be started. The ‘chkconfig –add’ command will also turn on level 2, 3, 4 and 5 automatically as shown below.

```
# chkconfig --list | grep iptables

# chkconfig --add iptables

# chkconfig --list | grep iptables
iptables        0:off   1:off   2:on    3:on    4:on
5:on    6:off
```

Note: "chkconfig –add" only adds an existing service to the list of startup. If the service doesn't exist, you should first install it before adding it to the system startup list. While this is pretty obvious, it is worth to mention it, as a newbie might make this mistake.

Remove a Service From Startup List

The following example shows that ip6tables services is configured for startup.

```
# chkconfig --list | grep ip6tables
ip6tables  0:off  1:off  2:off  3:on  4:off  5:off  6:off
```

To remove it from the startup list, use the –del option as shown below.

```
# chkconfig --del ip6tables

# chkconfig --list | grep ip6tables
```

Turn-on or Turn-off a Service for a Selected Run Level

Sometimes you might not want to delete the whole service. Instead, you might just want to turn the flag on or off for a particular run level (for a particular service).

The following example will turn off nfserver service for level 5

```
# chkconfig --level 5 nfsserver off
```

You can also combine multiple levels. The following example will turn off nfsserver for both level 3 and 5.

```
# chkconfig --level 35 nfsserver off
```

Script Files under rc.d Subdirectories

Whenever you add or remove a service from chkconfig control, it does the following under the /etc/rc.d sub-directories.

When chkconfig –add command is executed, it creates a symbolic link file to start and stop the service under corresponding rc directory.

When chkconfig –del command is executed, it removes the symbolic link file from the corresponding rc directory.

The following example shows that xinetd is enabled for both run level 3 and 5.

So, xinetd will have two files under rc3.d directory, and two files under rc5.d directory. The file that starts with K is used during shutdown (K stands for kill). The file that starts with S is used during startup (S stands for start).

```
# chkconfig --list | grep xinetd
xinetd   0:off  1:off  2:off  3:on   4:off  5:on   6:off
xinetd based services:

# cd /etc/rc.d/rc3.d
# ls | grep xinetd
K08xinetd
S14xinetd

# cd /etc/rc.d/rc5.d
```

```
# ls | grep xinetd
K08xinetd
S14xinetd
```

rcx.d Directory Changes for Add Operation

When you add a new service to chkconfig control, the default run levels for that service will be turned on automatically, and files will be created under the corresponding rcx directories.

For example, if the nfsserver service doesn't exist in the chkconfig control, no nfsserver service startup files would be present under /etc/rc.d/rc*.d directories as shown below.

```
# chkconfig  --list | grep nfsserver
nfsserver  0:off  1:off  2:off  3:off  4:off  5:off  6:off

# ls /etc/rc.d/rc3.d | grep nfsserver

# ls /etc/rc.d/rc5.d | grep nfsserver
```

After you add the nfsserver service, you'll see the symbolic links under these directories.

```
# chkconfig --add nfsserver
nfsserver  0:off  1:off  2:off  3:on   4:off  5:on   6:off

# cd  /etc/rc.d/rc3.d
# ls -l | grep nfsserver
lrwxrwxrwx 1 root root 12 2011-06-18 00:52 K08nfsserver ->
../nfsserver
lrwxrwxrwx 1 root root 12 2011-06-18 00:52 S14nfsserver ->
../nfsserver
```

```
# cd /etc/rc.d/rc5.d
# ls -l | grep nfsserver
lrwxrwxrwx 1 root root 12 2011-06-18 00:52 K08nfsserver ->
../nfsserver
lrwxrwxrwx 1 root root 12 2011-06-18 00:52 S14nfsserver ->
../nfsserver
```

When you turn off the service either using –del option or –level option, the corresponding symbolic link file under rcx.d directory will be deleted as shown below.

```
# chkconfig --level 5 nfsserver off

# ls /etc/rc.d/rc5.d  | grep nfsserver
```

Any Questions?

Discuss it here: 7 Linux chkconfig Command Examples – Add, Remove, View, Change Services

Hack 69. How to Setup Anacron

Anacron is the cron for desktops and laptops.

Anacron does not expect the system to be running 24 x 7 like a server.

When you want a background job to be executed automatically on a machine that is not running 24 x 7, you should use anacron.

For example, if you have a backup script scheduled everyday at 11 PM as a regular cron job, and if your laptop is not up at 11 PM, your backup job will not be executed.

However, if you have the same job scheduled in anacron, you can be sure that it will be executed once the laptop come back up.

Anacrontab Format

Just like how cron has /etc/crontab, anacron has /etc/anacrontab.

/etc/anacrontab file has the anacron jobs mentioned in the following format.

```
period   delay   job-identifier   command
```

Field 1 is Recurrence period: This is a numeric value that specifies the number of days.

- 1 – daily
- 7 – weekly
- 30 – monthly
- N – This can be any numeric value. N indicates number of days
- Note: You can also use '@monthly' for a job that needs to be executed monthly.

Field 2 is Delay: This indicates the delay in minutes. i.e X number of minutes anacron should wait before executing the job after the the machine starts.

Field 3 is Job identifier: It is the name for the job's timestamp file. It should be unique for each job. This will be available as a file under the /var/spool/anacron directory. This file will contain a single line that indicates the last time when this job was executed.

```
# ls -1 /var/spool/anacron/
test.daily
cron.daily
cron.monthly
cron.weekly
```

```
# cat /var/spool/anacron/test.daily
20110507
```

Field 4 is command: Command or shell script that needs to be executed.

Just like shell scripts, comments inside anacrontab file starts with #

Anacron Example

The following example executes the /home/sathiya/backup.sh script once in every 7 days.

On the day when the backup.sh job is supposed to executed, if the system is down for some reason, anacron will execute the backup.sh script 15 minutes after the system comes back up (without having to wait for another 7 days).

```
# cat /etc/anacrontab
7       15      test.daily      /bin/sh
/home/sathiya/backup.sh
```

START_HOURS_RANGE and RANDOM_DELAY

The above example indicates that the backup.sh script should be executed every day, with a delay of 15 mins. i.e When the laptop was started, executed it only after 15 minutes.

What happens when the laptop or desktop was not shutdown? When does the job gets executed? This is specified by the START_HOURS_RANGE environment variable in the /etc/anacrontab file.

By default this is set to 3-22 in the file. This indicates the time range from 3 a.m to 10 p.m.

```
# grep START /etc/anacrontab
START_HOURS_RANGE=3-22
```

On top of the user defined delay specified in the 2nd field of the /etc/anacrontab file, anacron also randomly adds x number of minutes. The x is defined by the RANDOM_DELAY variable in the /etc/anacrontab file.

By default this is set to 45 in the file. This means that anacron will add x minutes (randomly picked from 0 and 45), and add this to the user defined delay.

```
# grep RANDOM /etc/anacrontab
RANDOM_DELAY=45
```

Cron Vs Anacron

Cron and anacron has its own advantages and disadvantages. Depending on your requirement, use one of them.

Cron	Anacron
Minimum granularity is minute (i.e Jobs can be scheduled to be executed every minute)	Minimum granularity is only in days
Cron job can be scheduled by any normal user (if not restricted by super user	Anacron can be used only by super user (but there are workarounds to make it usable by normal user)
Cron expects system to be running 24 x 7. If a job is scheduled, and system is down during that time, job is not executed.	Anacron doesn't expect system to be running 24 x 7. If a job is scheduled, and system is down during that time, it start the jobs when the system comes back up.
Ideal for servers	Ideal for desktops and laptops

Use cron when a job has to be executed at a particular hour and minute	Use anacron when a job has to be executed irrespective of hour and minute

Any Questions?

Discuss it here: Cron Vs Anacron: How to Setup Anacron on Linux (With an Example)

Hack 70. IPTables Rules Examples

Delete Existing Rules

Before you start building new set of rules, you might want to clean-up all the default rules, and existing rules. Use the iptables flush command as shown below to do this.

```
iptables -F
(or)
iptables --flush
```

Set Default Chain Policies

The default chain policy is ACCEPT. Change this to DROP for all INPUT, FORWARD, and OUTPUT chains as shown below.

```
iptables -P INPUT DROP
iptables -P FORWARD DROP
iptables -P OUTPUT DROP
```

When you make both INPUT, and OUTPUT chain's default policy as DROP, for every firewall rule requirement you have, you should define two rules. i.e one for incoming and one for outgoing.

In all our examples below, we have two rules for each scenario, as we've set DROP as default policy for both INPUT and OUTPUT chain.

If you trust your internal users, you can omit the last line above. i.e Do not DROP all outgoing packets by default. In that case, for every firewall rule requirement you have, you just have to define only one rule. i.e define rule only for incoming, as the outgoing is ACCEPT for all packets.

Block a Specific ip-address

Before we proceed further will other examples, if you want to block a specific ip-address, you should do that first as shown below. Change the "x.x.x.x" in the following example to the specific ip-address that you like to block.

```
BLOCK_THIS_IP="x.x.x.x"
iptables -A INPUT -s "$BLOCK_THIS_IP" -j DROP
```

This is helpful when you find some strange activities from a specific ip-address in your log files, and you want to temporarily block that ip-address while you do further research.

You can also use one of the following variations, which blocks only TCP traffic on eth0 connection for this ip-address.

```
iptables -A INPUT -i eth0 -s "$BLOCK_THIS_IP" -j DROP
iptables -A INPUT -i eth0 -p tcp -s "$BLOCK_THIS_IP" -j
DROP
```

Allow ALL Incoming SSH

The following rules allow ALL incoming ssh connections on eth0 interface.

```
iptables -A INPUT -i eth0 -p tcp --dport 22 -m state
--state NEW,ESTABLISHED -j ACCEPT
iptables -A OUTPUT -o eth0 -p tcp --sport 22 -m state
--state ESTABLISHED -j ACCEPT
```

Allow Incoming SSH only from a Specific Network

The following rules allow incoming ssh connections only from 192.168.100.X network.

```
iptables -A INPUT -i eth0 -p tcp -s 192.168.100.0/24
--dport 22 -m state --state NEW,ESTABLISHED -j ACCEPT

iptables -A OUTPUT -o eth0 -p tcp --sport 22 -m state
--state ESTABLISHED -j ACCEPT
```

In the above example, instead of /24, you can also use the full subnet mask. i.e “192.168.100.0/255.255.255.0″.

Additional IPTables Examples:

25 Most Frequently Used Linux IPTables Rules Examples

IPTables Tables, Chains, Rules Fundamentals

How to Add Firewall Rules

Incoming and Outgoing Rule Examples

Chapter 9: Install Packages

Hack 71. Yum Command Examples

Installing, removing, and updating packages is a typical activity on Linux. Most of the Linux distributions provides some kind of package manager utility. For example, apt-get, dpkg, rpm, yum, etc.

On some Linux distributions, yum is the default package manager.

Yum stands for Yellowdog Updater Modified.

Install a package using yum install

To install a package, do 'yum install packagename'. This will also identify the dependencies automatically and install them.

The following example installs postgresql package.

```
# yum install postgresql.x86_64
Resolving Dependencies
Install       2 Package(s)
Is this ok [y/N]: y

Running Transaction
Installing : postgresql-libs-9.0.4-5.fc15.x86_64  1/2
Installing : postgresql-9.0.4-5.fc15.x86_64       2/2
```

By default 'yum install', will prompt you to accept or decline before installing the packages. If you want yum to install automatically without prompting, use -y option as shown below.

```
# yum -y install postgresql.x86_64
```

Uninstall a package using yum remove

To remove a package (along with all its dependencies), use ‘yum remove package’ as shown below.

```
# yum remove  postgresql.x86_64
Package postgresql.x86_64 0:9.0.4-5.fc15 will be erased
Is this ok [y/N]: y

Running Transaction
  Erasing    : postgresql-9.0.4-5.fc15.x86_64       1/1
```

Upgrade an existing package using yum update

If you have a older version of a package, use ‘yum update package’ to upgrade it to the latest current version. This will also identify and install all required dependencies.

```
# yum update postgresql.x86_64
```

Search for a package to be installed using yum search

If you don’t know the exact package name to be installed, use ‘yum search keyword’, which will search all the packages that matches the ‘keyword’ and display it.

The following examples searches the yum repository for all the packages that matches the keyword ‘firefox’ and lists the available packages.

```
# yum search firefox
Loaded plugins: langpacks, presto, refresh-packagekit
============== N/S Matched: firefox ======================
firefox.x86_64 : Mozilla Firefox Web browser
gnome-do-plugins-firefox.x86_64
mozilla-firetray-firefox.x86_64
mozilla-adblockplus.noarch : Mozilla Firefox extension
mozilla-noscript.noarch : Mozilla Firefox extension
```

Name and summary matches only, use "search all" for everything.

Display additional information about a package using yum info

Once you search for a package using yum search, you can use ‘yum info package’ to view additional information about the package.

The following examples displays additional information about the samba-common package.

```
# yum info samba-common.i686
Loaded plugins: langpacks, presto, refresh-packagekit
Available Packages
Name        : samba-common
Arch        : i686
Epoch       : 1
Version     : 3.5.11
Release     : 71.fc15.1
Size        : 9.9 M
Repo        : updates
Summary     : Files used by both Samba servers and clients
URL         : http://www.samba.org/
License     : GPLv3+ and LGPLv3+
Description : Samba-common provides files necessary for
both the server and client
```

Additional YUM Examples:

15 Linux Yum Command Examples – Install, Uninstall, Update Packages

Hack 72. RPM Command Examples

RPM command is used for installing, uninstalling, upgrading, querying, listing, and checking RPM packages on your Linux system.

RPM stands for Red Hat Package Manager.

With root privilege, you can use the rpm command with appropriate options to manage the RPM software packages.

Let us take an rpm of Mysql Client and run through all our examples.

Installing a RPM package Using rpm -ivh

RPM filename has packagename, version, release and architecture name.

For example, In the MySQL-client-3.23.57-1.i386.rpm file:

- MySQL-client – Package Name
- 3.23.57 – Version
- 1 – Release
- i386 – Architecture

When you install a RPM, it checks whether your system is suitable for the software the RPM package contains, figures out where to install the files located inside the rpm package, installs them on your system, and adds that piece of software into its database of installed RPM packages.

The following rpm command installs Mysql client package.

```
# rpm -ivh  MySQL-client-3.23.57-1.i386.rpm
Preparing...########################################### [100%]
   1:MySQL-client########################################### [100%]
```

rpm command and options

- -i : install a package
- -v : verbose
- -h : print hash marks as the package archive is unpacked.

You can also use dpkg on Debian, pkgadd on Solaris, depot on HP-UX to install packages.

Query all the RPM Packages using rpm -qa

You can use rpm command to query all the packages installed in your system.

```
# rpm -qa
cdrecord-2.01-10.7.el5
bluez-libs-3.7-1.1
setarch-2.0-1.1
.
.
```

- -q query operation
- -a queries all installed packages

To identify whether a particular rpm package is installed on your system, combine rpm and grep command as shown below. Following command checks whether cdrecord package is installed on your system.

```
# rpm -qa | grep 'cdrecord'
```

Query a Particular RPM Package using rpm -q

The above example lists all currently installed package. After installation of a package to check the installation, you can query a particular package and verify as shown below.

```
# rpm -q MySQL-client
MySQL-client-3.23.57-1
```

```
# rpm -q MySQL
package MySQL is not installed
```

Note: To query a package, you should specify the exact package name. If the package name is incorrect, then rpm command will report that the package is not installed.

Query RPM Packages in a various format using rpm –queryformat

Rpm command provides an option –queryformat, which allows you to give the header tag names, to list the packages. Enclose the header tag with in {}.

```
# rpm -qa --queryformat '%{name-%{version}-%{release} %
{size}\n'
cdrecord-2.01-10.7 12324
bluez-libs-3.7-1.1 5634
setarch-2.0-1.1 235563
.
.
```

Which RPM package does a file belong to? – Use rpm -qf

Let us say, you have list of files and you would want to know which package owns all these files. rpm command has options to achieve this.

The following example shows that /usr/bin/mysqlaccess file is part of the MySQL-client-3.23.57-1 rpm.

```
# rpm -qf /usr/bin/mysqlaccess
MySQL-client-3.23.57-1
```

Locate documentation of a package that owns file using rpm -qdf

Use the following to know the list of documentations, for a package that owns a file. The following command, gives the location of all the manual pages related to mysql package.

```
# rpm -qdf /usr/bin/mysqlaccess
/usr/share/man/man1/mysql.1.gz
/usr/share/man/man1/mysqlaccess.1.gz
/usr/share/man/man1/mysqladmin.1.gz
/usr/share/man/man1/mysqldump.1.gz
/usr/share/man/man1/mysqlshow.1.gz
```

- -d : refers documentation.

Information about Installed RPM Package using rpm -qip

rpm command provides a lot of information about the installed packages.

```
# rpm -qip MySQL-client-3.23.57-1.i386.rpm
Name        : MySQL-client Relocations: (not relocatable)
Version     : 3.23.57      Vendor: MySQL AB
Release     : 1            Build Date: Mon 09 Jun 2003
Install Date:              Build Host: build.mysql.com
Group       : Applications/Databases
Size        : 5305109      License: GPL / LGPL
Signature   : (none)
Packager    : Lenz Grimmer
URL         : http://www.mysql.com/
Summary     : MySQL - Client
Description : This package is a standard MySQL client.
```

- -i : view information about an rpm
- -p : specify a package name

List all the Files in a Package using rpm -qlp

To list the content of a RPM package, use the following command, which will list out the files without extracting into the local directory folder.

```
$ rpm -qlp ovpc-2.1.10.rpm
/usr/bin/mysqlaccess
/usr/bin/mysqldata
/usr/bin/mysqlperm
.
.
/usr/bin/mysqladmin
```

- q : query the rpm file
- l : list the files in the package
- p : specify the package name

List the Dependency Packages using rpm -qRP

To view the list of packages on which this package depends,

```
# rpm -qRp MySQL-client-3.23.57-1.i386.rpm
/bin/sh
/usr/bin/perl
```

Additional RPM Examples:

RPM Command: 15 Examples to Install, Uninstall, Upgrade, Query RPM Packages

Hack 73. apt-* Command Examples

Debian based systems (including Ubuntu) uses apt-* commands for managing packages from the command line.

This hack, using Apache 2 installation as an example, explains how to use apt-* commands to view, install, remove, or upgrade packages.

apt-cache search: Search Repository Using Package Name

If you are installing Apache 2, you may guess that the package name is apache2. To verify whether it is a valid package name, you may want to search the repository for that particular package name as shown below.

The following example shows how to search the repository for a specific package name.

```
$ apt-cache search ^apache2$
apache2 - Apache HTTP Server metapackage
```

dpkg -l: Is the Package Already Installed?

Before installing a package, you may want to make sure it is not already installed as shown below using dpkg -l command.

```
$ dpkg -l | grep -i apache
```

apt-get install: Install a Package

Finally, install the package using “apt-get install” as shown below.

```
$ sudo apt-get install apache2
[sudo] password for ramesh:

The following NEW packages will be installed:
  apache2 apache2-mpm-worker apache2-utils
  apache2.2-common libapr1 libaprutil1 libpq5

0 upgraded, 7 newly installed, 0 to remove and 26 not
upgraded.
```

apt-get remove: Delete a Package

Use “apt-get purge” or “apt-get remove” to delete a package as shown below.

```
$ sudo apt-get purge apache2

(or)

$ sudo apt-get remove apache2
```

Additional apt-* Command Examples:

How To Manage Packages Using apt-get, apt-cache, apt-file and dpkg Commands (With 13 Practical Examples)

Hack 74. Install from Source

Sometimes you might have to download the source code of an application and install it from the source.

This hack explains how to install from source. You should always refer to the INSTALL or README file that is located in the downloaded application package for any application specific installation instructions.

Download and unzip the package

Typically the source code you download will be compressed in a *.tar.gz or *.tar.bz2 format.

If the source code you've downloaded is in the format application.tar.gz, use the following command to uncompress it .

```
tar xvfz application.tar.gz
```

If the source code you've downloaded is in the format application.tar.bz2, use the following command to uncompress it.

```
tar xvfj application.tar.bz2
```

Configure

Once you uncompress the source tar file, it will create a subdirectory in the name of the application. CD to this directory.

```
cd application
```

Do a ./configure --help which will display all application specific configuration options that are available to you.

```
./configure --help
```

In most cases, you can just do ./configure which will use all default values to perform the configuration. This will perform necessary pre-req checks. This will also generate the Makefile required for the installation.

```
./configure
```

Make and Install

Make command will use the Makefile created from the above step and create the application binary executable.

```
make
```

Finally, do 'make install' which will install the application in the appropriate location.

```
make install
```

Chapter 10: LAMP Stack

Hack 75. Install Apache 2 with SSL

This hack provides step by step instructions on how to install Apache 2 with mod_ssl.

I prefer to install Apache from source, as it gives me more flexibility on exactly what modules I want to enable or disable, and I can also upgrade or apply patch immediately after it is released by the Apache foundation.

Download Apache

Download Apache from httpd.apache.org. The current stable release is 2.2.17.

Once you get the direct URL to download the latest stable version of Apache, use wget as shown below to download it directly to you server.

```
cd ~
wget http://www.eng.lsu.edu/mirrors/apache//httpd/httpd-2.2.17.tar.gz
tar xvfz httpd-2.2.17.tar.gz
```

Install Apache with SSL/TLS

View all available Apache installation and configuration options as shown below.

```
cd httpd-2.2.17
./configure --help
```

To install an Apache module, you would typically say –enable-{module-name}. For example, to install SSL with Apache, it is –enable-ssl. To install ldap module, it is –enable-ldap.

To uninstall any default module that comes with Apache, you would typically say –disable-{module-name}. For example, to disable basic authentication in Apache, it is –disable-auth-basic

In this example, we will install Apache with all default modules, with addition of –enable-ssl (to install mod_ssl for SSL support), and –enable-so, which helps to load modules in Apache during run-time via the Dynamic Shared Object (DSO) mechanism, rather than requiring a recompilation.

```
./configure --enable-ssl --enable-so
make
make install
```

Note: By default the above installs Apache under /usr/local/apache2. If you like to change this location, use –prefix option in the ./configure.

Enable SSL in httpd.conf

Apache configuration file httpd.conf is located under /usr/local/apache2/conf.

Uncomment the httpd-ssl.conf Include line in the /usr/local/apache2/conf/httpd.conf file.

```
# vi /usr/local/apache2/conf/httpd.conf
Include conf/extra/httpd-ssl.conf
```

View the httpd-ssl.conf to review all the default SSL configurations. For most cases, you don’t need to modify anything in this file.

```
vi /usr/local/apache2/conf/extra/httpd-ssl.conf
```

The SSL certificate and key are required before we start the Apache. The server.crt and server.key file mentioned in the httpd-ssl.conf needs to be created before we move forward.

```
 # egrep 'server.crt|server.key' httpd-ssl.conf
SSLCertificateFile "/usr/local/apache2/conf/server.crt"
SSLCertificateKeyFile "/usr/local/apache2/conf/server.key"
```

Create server.crt and server.key file

First, Generate the server.key using openssl.

```
cd ~
openssl genrsa -des3 -out server.key 1024
```

The above command will ask for the password. Make sure to remember this password. You need this while starting your Apache later.

If you don't provide a password, you'll get the following error message.

```
2415:error:28069065:lib(40):UI_set_result:result too
small:ui_lib.c:849:You must type in 4 to 8191 characters
```

Next, generate a certificate request file (server.csr) using the above server.key file.

```
openssl req -new -key server.key -out server.csr
```

Finally, generate a self signed ssl certificate (server.crt) using the above server.key and server.csr file.

```
openssl x509 -req -days 365 -in server.csr -signkey
server.key -out server.crt
```

For more details refer to: How To Generate SSL Key, CSR and Self Signed Certificate For Apache

Copy the server.key and server.crt

Copy the server.key and server.crt file to appropriate Apache configuration directory location.

```
cd ~
cp server.key /usr/local/apache2/conf/
cp server.crt /usr/local/apache2/conf/
```

Start the apache and verify SSL

Start the Apache as shown below.

```
/usr/local/apache2/bin/apachectl start
```

This will prompt you to enter the password for your private key.

```
Apache/2.2.17 mod_ssl/2.2.17 (Pass Phrase Dialog)
Server www.example.com:443 (RSA)
Enter pass phrase:

OK: Pass Phrase Dialog successful.
```

By default Apache SSL runs on 443 port. Open a web browser and verify that you can access your Apache using https://{your-ip-address}

Any Questions?

Discuss it here: How To Install Apache 2 with SSL on Linux (with mod_ssl, openssl)

Additional Apache Install Examples:

How To Generate SSL Key, CSR and Self Signed Certificate For Apache

Install Apache 2 from Source on Linux

Hack 76. Install PHP from Source

All Linux distributions comes with PHP. However, it is recommended to download latest PHP source code, compile and install on Linux. This will make it easier to upgrade PHP on an ongoing basis immediately after a new patch or release is available for download from PHP. This hack explains how to install PHP5 from source on Linux.

Prerequisites

Apache web server should already be installed. Refer to my previous post on How to install Apache 2 on Linux. If you are planning to use PHP with MySQL, you should have My SQL already installed.

Download PHP

Download the latest source code from PHP Download page. Current stable release is 5.2.6. Move the source to /usr/local/src and extract is as shown below.

```
# bzip2 -d php-5.2.6.tar.bz2
# tar xvf php-5.2.6.tar
```

Install PHP

View all configuration options available for PHP using ./configure –-help (two hyphen in front of help). The most commonly used option is –-prefix={install-dir-name} to install PHP on a user defined directory.

```
# cd php-5.2.6
# ./configure --help
```

In the following example, PHP will be compiled and installed under the default location /usr/local/lib with Apache configuration and MySQL support.

```
# ./configure --with-apxs2=/usr/local/apache2/bin/apxs
--with-mysql
# make
# make install
# cp php.ini-dist /usr/local/lib/php.ini
```

Configure httpd.conf for PHP

Modify the /usr/local/apache2/conf/httpd.conf to add the following:

```
<FilesMatch "\.ph(p[2-6]?|tml)$">
SetHandler application/x-httpd-php
</FilesMatch>
```

Make sure the httpd.conf has the following line that will get automatically inserted during the PHP installation process.

```
LoadModule php5_module modules/libphp5.so
```

Restart the apache as shown below:

```
# /usr/local/bin/apache2/apachectl restart
```

Verify PHP Installation

Create a test.php under /usr/local/apache2/htdocs with the following content

```
# vi test.php
<?php phpinfo(); ?>
```

Go to http://local-host/test.php , which will show a detailed information about all the PHP configuration options and PHP modules installed on the system.

Trouble shooting during installation

Error 1: configure: error: xml2-config not found:

While performing the ./configure during PHP installation, you may get the following error:

```
# ./configure --with-apxs2=/usr/local/apache2/bin/apxs
--with-mysql
Configuring extensions
checking whether to enable LIBXML support... yes
checking libxml2 install dir... no
checking for xml2-config path...
configure: error: xml2-config not found. Please check your
libxml2 installation.
```

Install thelibxml2-devel and zlib-devel as shown below to the fix this issue.

```
# rpm -ivh /home/downloads/linux-iso/libxml2-devel-2.6.26-
2.1.2.0.1.i386.rpm /home/downloads/linux-iso/zlib-devel-
1.2.3-3.i386.rpm
Preparing...########################################## [100%]
1:zlib-devel########################################## [ 50%]
2:libxml2-devel########################################## [100%]
```

Error 2: configure: error: Cannot find MySQL header files.

While performing the ./configure during PHP installation, you may get the following error:

```
# ./configure --with-apxs2=/usr/local/apache2/bin/apxs
--with-mysql
checking for MySQL UNIX socket location...
/var/lib/mysql/mysql.sock
configure: error: Cannot find MySQL header files under
yes. Note that the MySQL client library is not bundled
anymore!
```

Install the MySQL-devel-community package as shown below to fix this issue.

```
# rpm -ivh /home/downloads/MySQL-devel-community-5.1.25-
0.rhel5.i386.rpm
Preparing...########################################### [100%]
1:MySQL-devel-community############################## [100%]
```

Any Questions?

Discuss it here: Instruction Guide to Install PHP5 from Source on Linux

Hack 77. Install MySQL

Most of the Linux distro comes with MySQL. If you want use MySQL, my recommendation is that you download the latest version of MySQL and install it yourself. Later you can upgrade it to the latest version when it becomes available. This hack explains how to install the latest free community edition of MySQL on Linux platform.

Download the latest stable release of MySQL

Download mySQL from mysql.com . Please download the community edition of MySQL for your appropriate Linux platform. I downloaded the "Red Hat Enterprise Linux 5 RPM (x86)". Make sure to download MySQL Server, Client and "Headers and libraries" from the download page.

- MySQL-client-community-5.1.25-0.rhel5.i386.rpm
- MySQL-server-community-5.1.25-0.rhel5.i386.rpm
- MySQL-devel-community-5.1.25-0.rhel5.i386.rpm

If you want to remove the existing default MySQL that came with the Linux distro , do the following.

Do not perform this on an system where the MySQL database is getting used by some application.

```
[local-host]# rpm -qa | grep -i mysql
mysql-5.0.22-2.1.0.1
mysqlclient10-3.23.58-4.RHEL4.1

[local-host]# rpm -e mysql --nodeps
warning: /etc/my.cnf saved as /etc/my.cnf.rpmsave
[local-host]# rpm -e mysqlclient10
```

Install the downloaded MySQL package

Install the MySQL Server and Client packages as shown below.

```
[local-host]# rpm -ivh MySQL-server-community-5.1.25-
0.rhel5.i386.rpm MySQL-client-community-5.1.25-
0.rhel5.i386.rpm
Preparing...########################################### [100%]
1:MySQL-client-community############################## [ 50%]
2:MySQL-server-community############################## [100%]
```

This will also display the following output and start the MySQL daemon automatically.

```
PLEASE REMEMBER TO SET A PASSWORD FOR THE MySQL root USER!


To do so, start the server, then issue the following
commands:
/usr/bin/mysqladmin -u root password 'new-password'
/usr/bin/mysqladmin -u root -h medica2 password 'new-
password'


Alternatively you can run:
/usr/bin/mysql_secure_installation
```

```
Starting MySQL.[  OK  ]
Giving mysqld 2 seconds to start
```

Install the “Header and Libraries” that are part of the MySQL-devel packages.

```
[local-host]# rpm -ivh MySQL-devel-community-5.1.25-
0.rhel5.i386.rpm
Preparing...########################################### [100%]
1:MySQL-devel-community  ########################### [100%]
```

Note: When I was compiling PHP with MySQL option from source on the Linux system, it failed with the following error. Installing the MySQL-devel-community package fixed this problem in installing PHP from source.

```
configure: error: Cannot find MySQL header files under
yes.
```

Note that the MySQL client library is not bundled anymore!

Perform post-install security activities on MySQL.

At a bare minimum you should set a password for the root user as shown below:

```
[local-user]# /usr/bin/mysqladmin -u root password
'My2Secure$Password'
```

The best option is to run the mysql_secure_installation script that will take care of all the typical security related items on the MySQL as shown below. On a high level this does the following items:

- Change the root password
- Remove the anonymous user
- Disallow root login from remote machines

- Remove the default sample test database

```
[local-host]# /usr/bin/mysql_secure_installation


NOTE: RUNNING ALL PARTS OF THIS SCRIPT IS RECOMMENDED FOR
ALL MySQL SERVERS IN PRODUCTION USE!  PLEASE READ EACH
STEP CAREFULLY!
Enter current password for root (enter for none):
OK, successfully used password, moving on...


Change the root password? [Y/n] Y
New password:
Re-enter new password:
Password updated successfully!
Reloading privilege tables.. ... Success!


Remove anonymous users? [Y/n] Y


Disallow root login remotely? [Y/n] Y


Remove test database and access to it? [Y/n] Y


Reload privilege tables now? [Y/n] Y


installation should now be secure.
Thanks for using MySQL!
```

Verify the MySQL installation:

You can check the MySQL installed version by performing mysql -V as shown below:

```
 [local-host]# mysql -V
```

```
mysql  Ver 14.14 Distrib 5.1.25-rc, for redhat-linux-gnu
(i686) using readline 5.1
```

Connect to the MySQL database using the root user and make sure the connection is successful.

```
[local-host]# mysql -u root -p
Enter password:
mysql>
```

Follows the steps below to stop and start MySQL

```
[local-host]# service mysql status
MySQL running (12588)                                  [  OK  ]

[local-host]# service mysql stop
Shutting down MySQL.                                   [  OK  ]

[local-host]# service mysql start
Starting MySQL.                                        [  OK  ]
```

Any Questions?

Discuss it here: Howto Install MySQL on Linux

Additional MySQL Install Examples:

How to Install MySQL Database Using Yum groupinstall

Hack 78. Install LAMP Stack

Installing LAMP stack using yum is very easy and takes only minutes. This is a good option for beginners who don't feel comfortable installing from source. Also, Installing LAMP stack using yum is a good choice, if you want to keep things simple and just use the default configuration.

Install Apache using Yum

```
# rpm -qa | grep httpd
```

If the above command did not return anything, install apache as shown below

```
# yum install httpd
```

Verify that Apache got installed successfully

```
# rpm -qa | grep -i http
httpd-tools-2.2.9-1.fc9.i386
httpd-2.2.9-1.fc9.i386
```

Enable httpd service to start automatically during system startup using chkconfig. Start the Apache as shown below.

```
# chkconfig httpd on

# service httpd start
Starting httpd:                    [  OK  ]
```

Install MySQL using Yum

Yum is very smart to identify all the dependencies and install those automatically. For example, while installing mysql-server using yum, it also automatically installs the depended mysql-libs, perl-DBI, mysql, perl-DBD-MySQL packages as shown below.

```
 # yum install mysql-server
```

Partial output of the above yum install mysql-server command:

```
Dependencies Resolved

Transaction Summary
================================================================
Install      5 Package(s)
Update       0 Package(s)
Remove       0 Package(s)

Total download size: 15 M
Is this ok [y/N]: y

Running Transaction
Installing     : mysql-libs                 [1/5]
Installing     : perl-DBI                   [2/5]
Installing     : mysql                      [3/5]
Installing     : perl-DBD-MySQL             [4/5]
Installing     : mysql-server               [5/5]

Complete!
```

Verify whether MySQL got installed properly.

```
# rpm -qa | grep -i mysql
php-mysql-5.2.6-2.fc9.i386
mysql-libs-5.0.51a-1.fc9.i386
mysql-server-5.0.51a-1.fc9.i386
perl-DBD-MySQL-4.005-8.fc9.i386
mysql-5.0.51a-1.fc9.i386
```

```
# mysql -V
mysql  Ver 14.12 Distrib 5.0.51a, for redhat-linux-gnu
(i386) using readline 5.0
```

Configure MySQL to start automatically during system startup.

```
# chkconfig mysqld on
```

Start MySQL service.

```
# service mysqld start
```

The first time when you start mysqld, it will give additional information message indicating to perform post-install configuration as shown below.

```
Initializing MySQL database:
Installing MySQL system tables... OK
Filling help tables... OK

To start mysqld at boot time you have to copy
support-files/mysql.server to the right place for your
system

PLEASE REMEMBER TO SET A PASSWORD FOR MySQL root USER !
Start the server, then issue the following commands:
/usr/bin/mysqladmin -u root password 'new-password'
/usr/bin/mysqladmin -u root -h dev-db password 'new-
password'

Alternatively you can run:
/usr/bin/mysql_secure_installation
```

```
Starting MySQL:
[  OK  ]
```

Perform MySQL post-installation activities

After the mysql installation, you can login to mysql root account without providing any password as shown below.

```
# mysql -u root
mysql>
```

To fix this problem, you need to assign a password to mysql root account as shown below. Execute mysql_secure_installation script, which performs the following activities:

- Assign the root password
- Remove the anonymous user
- Disallow root login from remote machines
- Remove the default sample test database

```
# /usr/bin/mysql_secure_installation
```

Partial output of mysql_secure_installation script:

```
Enter current password for root (enter for none):
OK, successfully used password, moving on...

Set root password? [Y/n] Y
New password: [Note: Enter the mysql root password here]
Re-enter new password:
Password updated successfully!

Remove anonymous users? [Y/n] Y
Disallow root login remotely? [Y/n] Y
Remove test database and access to it? [Y/n] Y
```

```
Reload privilege tables now? [Y/n] Y
... Success!
```

Verify the MySQL post-install activities. Now root access without password is denied.

```
# mysql -u root
ERROR 1045 (28000):Access denied for user
'root'@'localhost'(using password:NO)
```

Test database is not available anymore.

```
# mysql -u root -p
Enter password:

mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
+--------------------+
2 rows in set (0.00 sec)
```

Install PHP using Yum

```
# yum install php
```

Partial output of yum install php:

```
Dependencies Resolved

Transaction Summary
```

```
============================================================
Install      3 Package(s)
Update       0 Package(s)
Remove       0 Package(s)

Total download size: 3.8 M
Is this ok [y/N]: y

Running Transaction
Installing     : php-common     [1/3]
Installing     : php-cli        [2/3]
Installing     : php            [3/3]

Complete!
```

Verify that php got installed successfully.

```
# rpm -qa | grep -i php
php-cli-5.2.6-2.fc9.i386
php-5.2.6-2.fc9.i386
php-common-5.2.6-2.fc9.i386
```

Install MySQL module for PHP.

```
# yum install php-mysql
```

Partial output of yum install php-mysql:

```
Dependencies Resolved
Transaction Summary
================================================
Install      2 Package(s)
```

```
Update       0 Package(s)
Remove       0 Package(s)

Total download size: 143 k
Is this ok [y/N]: y

Running Transaction
Installing     : php-pdo                 [1/2]
Installing     : php-mysql               [2/2]

Complete!
```

If you need additional PHP modules, install them using yum as shown below.

```
# yum install php-common php-mbstring php-mcrypt php-devel
php-xml php-gd
```

Any Questions?

Discuss it here: How To Install Or Upgrade LAMP Using Yum

Hack 79. Install XAMPP

To run various open source applications you might have to install Apache, MySQL, PHP, and Perl (or some combination of these). For those who have difficulties installing and configuring these separately, XAMPP might be helpful.

XAMPP is Apache distribution that contains MySQL, PHP and Perl. You really don't need to worry about configuring MySQL, PHP, or Perl for Apache. Just install XAMPP, and everything is already pre-built and ready to go. It is that easy!

Also, XAMPP is available for Linux, Windows, Mac OS, and Solaris.

Download XAMPP

Download XAMPP from xampp in sourceforge.

Install XAMPP

Extract the tar file under /opt directory.

```
# cd /opt
# tar xvzf xampp-linux-1.7.3a.tar.gz
```

Start or stop XAMPP

Start xampp. It starts all the xampp services.

```
# /opt/lampp/lampp start
```

Stop xampp. It stops all the xampp services.

```
# /opt/lampp/lampp stop
```

Start particular XAMPP service

Start a particular service by specifying the name of the service along with start. For example, following starts only Apache (along with PHP).

```
# /opt/lampp/lampp startapache
XAMPP: Starting Apache with SSL (and PHP5)...
Note: Use startmysql in the argument to start only mysql
```

Stop particular service

Similar to start, you can also stop a particular server. For example, following stops only MySQL.

```
# /opt/lampp/lampp stopmysql
XAMPP: Stopping MySQL...
```

XAMPP Configuration file

Configuration files are available under /opt/lampp/etc/. Some of the xampp config files are httpd.conf, my.cnf, php.ini.

Any Questions?

Discuss it here: XAMPP: Easy Apache, MySQL, PHP, Perl Install

Hack 80. Secure Your Apache Web Server

If you are a sysadmin, you should secure your Apache web server by following the tips mentioned in this hack.

Run Apache as separate user and group

By default, apache might run as nobody or daemon. It is good to run apache in its own non-privileged account. For example: apache.

Create apache group and user.

```
groupadd apache
useradd -d /usr/local/apache2/htdocs -g apache -s
/bin/false apache
```

Modify the httpd.conf, and set User and Group appropriately.

```
# vi httpd.conf
User apache
Group apache
```

After this, if you restart apache, and do ps -ef, you'll see that the apache is running as "apache" (Except the 1st httpd process, which will always run as root).

```
# ps -ef | grep -i http | awk '{print $1}'
root
apache
apache
apache
apache
apache
```

Restrict access to root directory (Use Allow and Deny)

Secure the root directory by setting the following in the httpd.conf

```
<Directory />
    Options None
    Order deny,allow
    Deny from all
</Directory>
```

In the above:

- Options None – Set this to None, which will not enable any optional extra features.
- Order deny,allow – This is the order in which the "Deny" and "Allow" directives should be processed. This processes the "deny" first and "allow" next.
- Deny from all – This denies request from everybody to the root directory. There is no Allow directive for the root directory. So, nobody can access it.

Set appropriate permissions for conf and bin directory

bin and conf directory should be viewed only by authorized users. It is good idea to create a group, and add all users who are allowed to view/modify the apache configuration files to this group.

Let us call this group: apacheadmin

Create the group.

```
groupadd apacheadmin
```

Allow access to bin directory for this group.

```
chown -R root:apacheadmin /usr/local/apache2/bin
chmod -R 770 /usr/local/apache2/bin
```

Allow access to conf directory for this group.

```
chown -R root:apacheadmin /usr/local/apache2/conf
chmod -R 770 /usr/local/apache2/conf
```

Add appropriate members to this group. In this example, both ramesh and john are part of apacheadmin

```
# vi /etc/group
apacheadmin:x:1121:ramesh,john
```

Disable Directory Browsing

If you don't do this, users will be able to see all the files (and directories) under your root (or any sub-directory).

For example, if they go to http://{your-ip}/images/ and if you don't have an index.html under images, they'll see all the image files (and the sub-directories) listed in the browser (just like a ls -1 output). From here, they

can click on the individual image file to view it, or click on a sub-directory to see its content.

To disable directory browsing, you can either set the value of Options directive to “None” or “-Indexes”. A – in front of the option name will remove it from the current list of options enforced for that directory.

Indexes will display a list of available files and sub-directories inside a directory in the browser (only when no index.html is present inside that folder). So, Indexes should not be allowed.

```
<Directory />
  Options None
  Order allow,deny
  Allow from all
</Directory>

(or)

<Directory />
  Options -Indexes
  Order allow,deny
  Allow from all
</Directory>
```

Don’t allow .htaccess

Using .htaccess file inside a specific sub-directory under the htdocs (or anywhere outside), users can overwrite the default apache directives. On certain situations, this is not good, and should be avoided. You should disable this feature.

You should not allow users to use the .htaccess file and override apache directives. To do this, set “AllowOverride None” in the root directory.

```
<Directory />
```

```
  Options None
  AllowOverride None
  Order allow,deny
  Allow from all
</Directory>
```

Additional Apache Hardening Examples:

10 Tips to Secure Your Apache Web Server on UNIX / Linux

Hack 81. Apachectl and Httpd Tips

After you have installed Apache2, if you want to use apachectl and httpd to it's maximum potential, you should go beyond using start, stop and restart. The 9 practical examples provided in this hack will help you to use apachectl and httpd very effectively.

Apachectl acts as SysV init script, taking arguments like start, stop, restart and status. It also acts as front-end to httpd command, by simply passing the command line arguments to httpd. So, all the commands you execute using apachectl, can also be executed directly by calling httpd.

Pass different httpd.conf filename to apachectl

Typically you'll modify the original httpd.conf to try out different Apache directives. If something doesn't work out, you'll revert back the changes. Instead of playing around with the original httpd.conf, copy it to a new httpd.conf.debug and use this new httpd.conf.debug file with Apache for testing purpose as shown below using option -f.

```
# apachectl -f conf/httpd.conf.debug

# httpd -k start -f conf/httpd.conf.debug
```

```
[Note: you can use either apachectl or httpd as shown
above]


# ps -ef | grep http
root   25080     1  0 23:26 00:00:00 /usr/sbin/httpd -f
conf/httpd.conf.debug
apache 25099 25080  0 23:28 00:00:00 /usr/sbin/httpd -f
conf/httpd.conf.debug
[Note: ps shows the httpd running with httpd.conf.debug
file]
```

Once you are satisfied with the changes and Apache runs without any problem with httpd.conf.debug, you can copy the changes to httpd.conf and start the Apache normally as shown below.

```
# cp httpd.conf.debug httpd.conf
# apachectl stop
# apachectl start


# ps -ef | grep httpd
root     25114     1  0 23:28 00:00:00 /usr/sbin/httpd -k
start
daemon   25115 25114  0 23:28 00:00:00 /usr/sbin/httpd -k
start
[Note: ps indicates that the httpd is running using the
default config file]
```

Use a temporary DocumentRoot without modifying httpd.conf

This is very helpful, when you are trying out different layout for your website and don't want to modify the original files under the default DocumentRoot. Take a copy of your original DocumentRoot directory (/var/www/html) to a new temporary DocumentRoot directory (/var/www/html_debug). Make all your changes under this temporary DocumentRoot directory (/var/www/html_debug) and start the Apache with this temporary directory as shown below using option -c.

```
# httpd -k start -c "DocumentRoot /var/www/html_debug/"
```

If you want to go back to original configuration using the default DocumentRoot (/var/www/html), simply restart the Apache as shown below.

```
# httpd -k stop
# apachectl start
```

Increase the LogLevel temporarily

While you are debugging an issue, you can change the LogLevel of the Apache temporarily, without modifying the LogLevel directive in the httpd.conf as shown below using option -e. In this example, the LogLevel is set to debug.

```
# httpd -k start -e debug
[Sun Aug 17 13:53:06 2008] [debug] mod_so.c(246): loaded
module auth_basic_module
[Sun Aug 17 13:53:06 2008] [debug] mod_so.c(246): loaded
module auth_digest_module
```

Possible values you can pass to option -e are: debug, info, notice, warn, error, crit, alert, emerg

Display the modules compiled inside Apache using option -l

```
# httpd -l
Compiled in modules:
core.c
prefork.c
http_core.c
mod_so.c
```

Display both static and dynamic module loaded by Apache

When you pass option -l, to httpd, it will display only the static modules. Passing option -M, will display both static and shared modules as shown below.

```
# httpd -M
Loaded Modules:
core_module (static)
mpm_prefork_module (static)
http_module (static)
so_module (static)
auth_basic_module (shared)
auth_digest_module (shared)
authn_file_module (shared)
authn_alias_module (shared)
Syntax OK
```

Show all accepted directives inside httpd.conf

This is like an extended help for httpd, which will display all the httpd.conf directives and the places where they are valid. For a specific directive, it tells all the possible values and where it can be used inside the httpd.conf. This can be very helpful, when you want to quickly know about a particular Apache directive.

```
# httpd -L
HostnameLookups (core.c)
"on" to enable, "off" to disable reverse DNS lookups, or
"double" to enable double-reverse DNS lookups
Allowed in *.conf anywhere


ServerLimit (prefork.c)
Maximum value of MaxClients for this run of Apache
Allowed in *.conf only outside <Directory>, <Files> or
<Location>
```

```
KeepAlive (http_core.c)
Whether persistent connections should be On or Off
Allowed in *.conf only outside <Directory>, <Files> or
<Location>


LoadModule (mod_so.c)
a module name and the name of a shared object file to load
it from
Allowed in *.conf only outside <Directory>, <Files> or
<Location>
```

Validate the httpd.conf after making changes

Use option -t to validate whether there are any issues with a specific Apache configuration file. In the example shown below, it displays that there is a problem at line 148 in the httpd.conf.debug. mod_auth_basicso is missing a . (period) before the so.

```
# httpd -t -f conf/httpd.conf.debug
httpd: Syntax error on line 148 of
/etc/httpd/conf/httpd.conf.debug:
Cannot load /etc/httpd/modules/mod_auth_basicso into
server:
/etc/httpd/modules/mod_auth_basicso: cannot open shared
object file: No such file or directory
Once you fix the issue, it will display Syntax OK.


# httpd -t -f conf/httpd.conf.debug
Syntax OK
```

Display the httpd build parameters

Use option -V (upper-case V), to display Apache version number and all the parameters that are used while building the Apache.

```
# httpd -V
Server version: Apache/2.2.9 (Unix)
```

```
Server built:   Jul 14 2008 15:36:56
Server's Module Magic Number: 20051115:15
Server loaded:  APR 1.2.12, APR-Util 1.2.12
Compiled using: APR 1.2.12, APR-Util 1.2.12
Architecture:   32-bit
Server MPM:     Prefork
threaded:     no
forked:     yes (variable process count)
Server compiled with....
-D APACHE_MPM_DIR="server/mpm/prefork"
-D APR_HAS_SENDFILE
-D HTTPD_ROOT="/etc/httpd"
-D SUEXEC_BIN="/usr/sbin/suexec"
-D DEFAULT_PIDLOG="logs/httpd.pid"
-D DEFAULT_SCOREBOARD="logs/apache_runtime_status"
-D DEFAULT_LOCKFILE="logs/accept.lock"
-D DEFAULT_ERRORLOG="logs/error_log"
-D AP_TYPES_CONFIG_FILE="conf/mime.types"
-D SERVER_CONFIG_FILE="conf/httpd.conf"
...
```

If you want display only the Apache version number, use the option -v (lower-case v) as shown below.

```
# httpd -v
Server version: Apache/2.2.9 (Unix)
Server built:   Jul 14 2008 15:36:56
```

Load a specific module only on demand.

Sometimes you may not want to load all the modules in the Apache. For e.g. You may want to load ldap related modules to Apache, only when you are testing LDAP. This can be achieved as shown below.

Modify the httpd.conf and add IfDefine directive called load-ldap (you can name this anything you want).

```
<IfDefine load-ldap>
LoadModule ldap_module modules/mod_ldap.so
LoadModule authnz_ldap_module modules/mod_authnz_ldap.so
</IfDefine>
```

When you are testing ldap and would like to Load the ldap related modules, pass the load-ldap to Option -D, as shown below:

```
# httpd -k start -e debug -Dload-ldap -f
/etc/httpd/conf/httpd.conf.debug
[Sun Aug 17 14:14:58 2008] [debug] mod_so.c(246): loaded
module ldap_module
[Sun Aug 17 14:14:58 2008] [debug] mod_so.c(246): loaded
module authnz_ldap_module
[Note: Pass -Dload-ldap, to load the ldap modules into
Apache]
```

```
# apachectl start
[Note: Start the Apache normally, if you don't want to
load the ldap modules.]
```

Any Questions?

Discuss it here: 9 Tips to Use Apachectl and Httpd like a Power User

Hack 82. Setup Apache Virtual Host Configuration

Uncomment httpd-vhosts.conf in httpd.conf

If you've installed Apache 2 from source, by default, the following line will be commented in the httpd.conf file. Uncomment this line.

```
# vi /usr/local/apache2/conf/httpd.conf
Include conf/extra/httpd-vhosts.conf
```

Setup virtual hosts

Modify the httpd-vhosts.conf as shown below to setup named-based virtual host setting for two hosts.

- NameVirtualHost *:80 – Indicates that all the name-based virtual hosts will be listening on the default port 80
- <VirtualHost *:80> </VirtualHost> – Enclose all the apache configuration parameters for each and every virtual host between these VirtualHost tags. Any apache directives can be used within the virtualhost container.
- In the following example, we are setting up virtual host for thegeekstuff.com and top5freeware.com listening on the same port 80. So, there will be two <VirtualHost *:80> </VirtualHost>, one for each website.
- When you go to thegeekstuff.com, the files under /usr/local/apache2/docs/thegeekstuff will be served by Apache; and the access_log and error_log for this site will go under /usr/local/apache2/logs/thegeekstuff

```
# vi /usr/local/apache2/conf/extra/httpd-vhosts.conf
NameVirtualHost *:80

<VirtualHost *:80>
    ServerAdmin ramesh@thegeekstuff.com
    DocumentRoot "/usr/local/apache2/docs/thegeekstuff"
    ServerName thegeekstuff.com
```

```
    ServerAlias www.thegeekstuff.com
    ErrorLog "logs/thegeekstuff/error_log"
    CustomLog "logs/thegeekstuff/access_log" common
</VirtualHost>

<VirtualHost *:80>
    ServerAdmin ramesh@top5freeware.com
    DocumentRoot "/usr/local/apache2/docs/top5freeware"
    ServerName top5freeware.com
    ServerAlias www.top5freeware.com
    ErrorLog "logs/top5freeware/error_log"
    CustomLog "logs/top5freeware/access_log" common
</VirtualHost>
```

Check VirtualHost Configuration Syntax

Verify virtual configuration syntax using “httpd -S” as shown below. When everything is setup properly, it just displays “Syntax OK”.

```
# /usr/local/apache2/bin/httpd -S
VirtualHost configuration:
Syntax OK
```

When something is not configured properly, it will display warning message, including “directory does not exit” message as shown below.

```
# /usr/local/apache2/bin/httpd -S
Warning: DocumentRoot
[/usr/local/apache2/docs/top5freeware] does not exist
Warning: ErrorLog [/usr/local/apache2/logs/thegeekstuff]
does not exist
Syntax OK
```

Restart the Apache and test

```
# /usr/local/apache2/bin/apachectl restart
```

Now, when you go to thegeekstuff.com (or www.thegeekstuff.com), the apache will serve the files from /usr/local/apache2/docs/thegeekstuff directory.

When you go to top5freeware.com (or www.top5freeware.com), the same apache running on the same server will serve the files from /usr/local/apache2/docs/top5freeware directory.

Just to reiterate, for the name-based virtual host to work properly, the DNS entry for both these websites should be pointing to the same external ip-address of the physical server where the Apache webserver is running.

Any Questions?

Discuss it here: How To Setup Apache Virtual Host Configuration (With Examples)

Hack 83. Rotate Apache Logs Files

This hack explains how to rotate the apache access_log and error_log files.

Add the following file to /etc/logrotate.d directory.

```
# vi /etc/logrotate.d/apache
/usr/local/apache2/logs/access_log
/usr/local/apache2/logs/error_log {
    size 100M
    compress
```

```
    dateext
    maxage 30
    postrotate
      /usr/bin/killall -HUP httpd
      ls -ltr /usr/local/apache2/logs | mail -s
"$HOSTNAME: Apache restarted and log files rotated"
ramesh@thegeekstuff.com
    endscript
}
```

Note: Refer to our logrotate tutorial (with 15 examples) that explains more details about how to use logrotate options.

In the above /etc/logrotate.d/apache example:

- size 100M – Once the access_log, and error_log reaches 100M, it will be rotated. You can also use 100k (for Kb), 100G (for GB). Instead of size, you can also rotate apache logs using frequency (daily, weekly, monthly).
- compress – Indicates that the rotated log file will be compressed. By default this uses gzip. So, the rotated file will have .gz extension.
- dateext - Appends the date in YYYYMMDD format to the rotated log files. i.e Instead of access_log.1.gz, it creates access_log-20110616.gz
- maxage - Indicates how long the rotated log files should be kept. In this example, it will be kept for 30 days.
- postrotate and endscript – Any commands enclosed between these two parameter will be executed after the log is rotated.

Important: Once you rotate the log files, you want apache to write the new log messages to the newly created access_log and error_log. So, you need to send the HUP signal to the apache as shown here. Make sure to do /usr/bin/killall -HUP httpd, which will restart the apache after rotating the log files (Read more about kill).

Also, you might want to send an email to yourself indicating that the log file is rotated, along with the output of ls -ltr command as the body of the email. i.e Add the following between “postrotate” and “endscript” option (after the killall command).

```
ls -ltr /usr/local/apache2/logs | mail -s "$HOSTNAME:
Apache restarted and log files rotated"
ramesh@thegeekstuff.com
```

The /etc/cron.daily/logrotate script runs everyday that will perform log rotate of all the files as specified in the /etc/logrotate.conf and all the file under /etc/logrotate.d directory.

After adding the above /etc/logrotate.d/apache file, for testing purpose, you can manually call the logrotate script as shown below.

```
# /etc/cron.daily/logrotate
```

Once the log files are rotated, do a ls to verify them. As we explained above, the rotated log files will be kept for 30 days.

```
# ls /usr/local/apache2/logs
access_log
error_log
access_log-20110716.gz
error_log-20110716.gz
```

Any Questions?

Discuss it here: How to Rotate Apache Log Files in Linux

Additional Logrotate Examples:

The Ultimate Logrotate Command Tutorial with 10 Examples

Chapter 11: Bash Scripting

Hack 84. Execution Sequence of .bash_* files

This hack explains the sequence in which the following files are executed:

- /etc/profile
- ~/.bash_profile
- ~/.bashrc
- ~/.bash_login
- ~/.profile
- ~/.bash_logout

Execution sequence for interactive login shell

Following pseudo code explains the sequence of execution of these files.

```
execute /etc/profile
IF ~/.bash_profile exists THEN
    execute ~/.bash_profile
ELSE
    IF ~/.bash_login exist THEN
        execute ~/.bash_login
    ELSE
        IF ~/.profile exist THEN
            execute ~/.profile
        END IF
    END IF
END IF
```

When you logout of the interactive shell, following is the sequence of execution:

```
 IF ~/.bash_logout exists THEN
    execute ~/.bash_logout
END IF
```

Please note that /etc/bashrc is executed by ~/.bashrc as shown below:

```
# cat ~/.bashrc
if [ -f /etc/bashrc ]; then
. /etc/bashrc
fi
```

Execution sequence for interactive non-login shell

While launching a non-login interactive shell, following is the sequence of execution:

```
IF ~/.bashrc exists THEN
    execute ~/.bashrc
END IF
```

Note: When a non-interactive shell starts up, it looks for ENV environment variable, and execute the file-name value mentioned in the ENV variable.

Test the sequence of execution

One of the ways to test the sequence of execution is by adding different PS1 values to these files and re-login to the shell and see which PS1 value got picked up by the Linux prompt. Also, earlier we discussed about how to use PS1 to make your Linux prompt both functional and stylish.

1. /etc/profile gets executed. Add following PS1 line to /etc/profile and re-login to make sure the Linux prompt changes to the PS1 value set inside the /etc/profile.

```
# grep PS1 /etc/profile
PS1="/etc/profile> "

[Note: re-login to see the prompt change as shown below]
Last login: Sat Sep 27 16:43:57 2008 from 192.168.1.2
/etc/profile>
```

Please make sure ~/.bash_profile doesn't have any PS1 for the above to work properly.

2. ~/.bash_profile gets executed: Add following PS1 to ~/.bash_profile, ~/.bash_login, ~/.profile and ~/.bashrc. Re-login to make sure the Linux prompt changes to the PS1 value set inside the ~/.bash_profile as shown below.

```
/etc/profile> grep PS1 ~/.bash_profile
export PS1="~/.bash_profile> "

/etc/profile> grep PS1 ~/.bash_login
export PS1="~/.bash_login> "

/etc/profile> grep PS1 ~/.profile
export PS1="~/.profile> "

/etc/profile> grep PS1 ~/.bashrc
export PS1="~/.bashrc> "

[Note: Upon re-login, it executed /etc/profile first and
~/.bash_profile next. So, it took the PS1 from
~/.bash_profile as shown below. It also did not execute
~/.bash_login, as ~/.bash_profile exists]
Last login: Sat Sep 27 16:48:11 2008 from 192.168.1.2
~/.bash_profile>
```

3. ~/.bash_login gets executed. Rename the .bash_profile to something else. Re-login to make sure the Linux prompt changes to the PS1 value set inside the ~/.bash_login as shown below.

```
~/.bash_profile> mv .bash_profile bash_profile_not_used


[Note: Upon re-login, it executed /etc/profile first.
Since it cannot find ~/.bash_profile, it executed
~/.bash_login]
Last login: Sat Sep 27 16:50:55 2008 from 192.168.1.2
~/bash_login>
```

4. ~/.profile gets executed. Rename the .bash_login to something else. Re-login to make sure the Linux prompt changes to the PS1 value set inside the ~/.profile as shown below.

```
~/.bash_login> mv .bash_login bash_login_not_used


[Note: Upon re-login, it executed /etc/profile first.
Since it cannot find ~/.bash_profile and ~/.bash_login, it
executed ~/.profile]
Last login: Sat Sep 27 16:56:36 2008 from 192.168.1.2
~/.profile>
```

5. ~/.bashrc gets executed for non-login shell testing. Executing “bash” at the command prompt will give another non-login shell, which will invoke .bashrc as shown below.

```
~/.profile> bash


[Note: This displays PS1 from .bashrc as shown below.]
~/.bashrc> exit


[Note: After exiting from non-login shell, we are back to
login shell]
```

```
~/.profile>
```

Any Questions?

Discuss it here: Execution sequence for .bash_profile, .bashrc, .bash_login, .profile and .bash_logout

Hack 85. Bash FOR Loops Using C Like Syntax

The second form of bash for loop is similar to the 'C' programming language for loop, which has three expressions (initialization, condition and update).

```
for (( expr1; expr2; expr3 ))
do
commands
done
```

- Before the first iteration, expr1 is evaluated. This is usually used to initialize variables for the loop.
- All the statements between do and done are executed repeatedly as long as the value of expr2 is TRUE.
- After each loop iteration, expr3 is evaluated. This is usually used to increment a loop counter.

The following examples show how to use this syntax in the bash for loop.

Loop using C-Style

Generate and display 5 random numbers using the bash C-style for loop:

```
$ cat for10.sh
for (( i=1; i <= 3; i++ ))
do
```

```
echo "Random number $i: $RANDOM"
done
$ ./for10.sh
Random number 1: 23320
Random number 2: 5070
Random number 3: 15202
```

Infinite Loop Using Bash For

When you don't provide the start, condition, and increment in a C-style for loop, it will execute forever. You need to press Ctrl-C to stop the loop.

```
$ cat for11.sh
i=1;
for (( ; ; ))
do
sleep $i
echo "Number: $((i++))"
done
```

Note: Don’t forget you will need to press Ctrl-C to break from this example:

```
$ ./for11.sh
Number: 1
Number: 2
Number: 3
```

Increment Two Values Using Comma in C-style for loop

In the bash c-style loop, in addition to incrementing the value that is used in the condition, you can also increment some other value or perform some other action. In both the initialization section and the increment section of the C-style for loop, you can use multiple statements separated with a comma. This example uses i for control and manipulates j separately:

```
$ cat for12.sh
for ((i=1, j=10; i <= 5 ; i++, j=j+5))
do
echo "Number $i: $j"
done

$ ./for12.sh
Number 1: 10
Number 2: 15
Number 3: 20
Number 4: 25
Number 5: 30
```

Additional Bash For Loop Examples:

12 Bash For Loop Examples for Your Linux Shell Scripting

Hack 86. Debug a Shell Script

To debug a shell script use set –xv inside the shell script at the top.

Shell script with no debug command:

```
$ cat filesize.sh
#!/bin/bash
for filesize in $(ls -l . | grep "^-" | awk '{print $5}')
do
  let totalsize=$totalsize+$filesize
done
echo "Total file size in current directory: $totalsize"
```

Output of Shell script with no debug command:

```
$ ./filesize.sh
Total file size in current directory: 652
```

Shell script with Debug command inside:

Add set –xv inside the shell script now to debug the output as shown below.

```
$ cat filesize.sh
#!/bin/bash
set -xv
for filesize in $(ls -l . | grep "^-" | awk '{print $5}')
do
  let totalsize=$totalsize+$filesize
done
echo "Total file size in current directory: $totalsize"
```

Output of Shell script with Debug command inside:

```
$ ./fs.sh
++ ls -l .
++ grep '^-'
++ awk '{print $5}'
+ for filesize in '$(ls -l . | grep "^-" | awk '\''{print $5}'\'')'
+ let totalsize=+178
+ for filesize in '$(ls -l . | grep "^-" | awk '\''{print $5}'\'')'
+ let totalsize=178+285
+ for filesize in '$(ls -l . | grep "^-" | awk '\''{print $5}'\'')'
+ let totalsize=463+189
+ echo 'Total file size in current directory: 652'
```

```
Total file size in current directory: 652
```

Execute Shell script with debug option:

Instead of giving the set –xv inside the shell script, you can also provide that while executing the shell script as shown below.

```
$ bash -xv filesize.sh
```

Hack 87. Quoting

echo statement without any special character.

```
$ echo The Geek Stuff
The Geek Stuff
```

Echo statement with a special character ; . semi-colon is a command terminator in bash. In the following example, “The Geek” works for the echo and “Stuff” is treated as a separate Linux command and gives command not found.

```
$ echo The Geek; Stuff
The Geek
-bash: Stuff: command not found
```

To avoid this you can add a \ in front of semi-colon, which will remove the special meaning of semi-colon and just print it as shown below.

```
$ echo The Geek\; Stuff
The Geek; Stuff
```

Single Quote

Use single quote when you want to literally print everything inside the single quote. Even the special variables such as $HOSTNAME will be print as $HOSTNAME instead of printing the name of the Linux host.

```
$ echo 'Hostname=$HOSTNAME ;  Current User=`whoami` ;
Message=\$ is USD'


Hostname=$HOSTNAME ;  Current User=`whoami` ; Message=\$
is USD
```

Double Quote

Use double quotes when you want to display the real meaning of special variables.

```
$ echo "Hostname=$HOSTNAME ;  Current User=`whoami` ;
Message=\$ is USD"


Hostname=dev-db ;  Current User=ramesh ; Message=$ is USD
```

Double quotes will remove the special meaning of all characters except the following:

- $ Parameter Substitution.
- ` Backquotes
- \$ Literal Dollar Sign.
- \´ Literal Backquote.
- \" Embedded Doublequote.
- \\ Embedded Backslashes.

Hack 88. Read Data File Fields Inside a Shell Script

This example shows how to read a particular field from a data-file and manipulate it inside a shell-script. For example, let us assume the employees.txt file is in the format of {employee-name}:{employee-id}:{department-name}, with colon delimited file as shown below.

```
$ cat employees.txt
Emma Thomas:100:Marketing
```

```
Alex Jason:200:Sales
Madison Randy:300:Product Development
Sanjay Gupta:400:Support
Nisha Singh:500:Sales
```

The following shell script explains how to read specific fields from this employee.txt file.

```
$ vi read-employees.sh
#!/bin/bash
IFS=:
echo "Employee Names:"
echo "---------------"
while read name empid dept
do
  echo "$name is part of $dept department"
done < ~/employees.txt
```

Assign execute privilege to the shell script and execute it.

```
$ chmod u+x read-employees.sh

$ ./read-employees.sh
Employee Names:
---------------
Emma Thomas is part of Marketing department
Alex Jason is part of Sales department
Madison Randy is part of Product Development department
Sanjay Gupta is part of Support department
Nisha Singh is part of Sales department
```

Chapter 12: System Monitoring and Performance

Hack 89. Free Command

free command displays all the necessary information about system physical (RAM) and swap memory.

```
Syntax: free [options]
```

What is the total RAM on my system?

In the example below, the total physical memory on this system is 1GB. The values displayed below are in KB.

```
# free
       total   used    free   shared  buffers  cached
Mem: 1034624   1006696 27928  0       174136   615892
-/+ buffers/cache:     216668      817956
Swap:2031608       0    2031608
```

What is the total memory on my system including RAM and Swap?

In the following command:

- option m displays the values in MB
- option t displays the “Total” line, which is sum of physical and swap memory values
- option o is to hide the buffers/cache line from the above example.

```
# free –mto
      total    used    free shared    buffers     cached
Mem:     1010   983      27      0        170        601
```

```
Swap:    1983     0    1983
Total:   2994   983    2011
```

Hack 90. Top Command

top command displays real time information about various performance metrics of the system such as CPU Load, Memory Usage, Processes list etc.

```
Syntax: top [options]
```

How to view my current system status including CPU usage?

Execute top without any option from the command line, which will display the output shown below. The top command output will keep displaying the real-time values, until you press “Control + c” or q to exit from the command output.

```
# top
top - 13:10:13 up 171 days, 20:21,  3 users,  load
average: 0.01, 0.05, 0.00


Tasks: 194 total,   1 running, 193 sleeping,   0 stopped,
0 zombie
Cpu(s):  0.6% us,  0.7% sy,  0.0% ni, 98.7% id,  0.0% wa,
0.0% hi,  0.0% si
Mem:   1034624k total,  1007420k used,       27204k free,
174540k buffers
Swap:  2031608k total,              0k used,  2031608k
free,    615904k cached


PID   USER    PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+
COMMAND
11912 apache    15     0 31828  13m  3916 S          1
0.2   0:46.35   httpd
```

```
19299  oracle    19     0  279m  18m   17m S          1
0.2   0:00.03   oracle
11398 jsmith     16     0  107m  28m  6404 S          0
0.4   0:03.07   perl
```

How to read the output of the top command shown above?

- Line 1 “top”, indicates that the system has been up and running for 171 days.
- Line 2 “Tasks”, displays the total number of processes along with a breakdown of running, sleeping, stopped and zombie processes count.
- Line 3 “Cpu(s)” displays the current CPU utilization of the system. In this example, CPU is 98.7% idle
- Line 4 “Mem” and line 5 “Swap” provides the memory information. This is the same information from the free command.
- The rest of the lines display all the active processes on the system, sorted default by CPU usage (%CPU column). i.e the most CPU intensive processes will be displayed on the top by default.

There are several command line options and interactive options available for top commands. Let us review couple of essential options for top command.

How to identify the most memory intensive processes?

While the output of the top command displayed, press F, which will display the following message and show all fields available for sorting, press n (which is for sorting the processes by Memory) and press enter. This will display the processes in the top output sorted by memory usage.

```
Current Sort Field:  K  for window 1:Def
```

Select sort field via field letter, type any other key to return

How to add additional fields (for e.g. CPU Time) to the top output?

While the top command is running, press f, which will display the following message and show all fields available for display, press l, which will add the CPU Time to the display columns in the top output.

```
Current Fields:  AEHIOQTWKNMbcdfgjplrsuvyzX  for window
1:Def
```

Toggle fields via field letter, type any other key to return

How to get the full path name and parameters of the running processes?

While the top command is running, press c, which will display full pathname of running processes as shown below in the command column. i.e Instead of httpd, it displays /usr/local/apache2/bin/httpd.

```
PID   USER    PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+
COMMAND
11912 apache    15     0 31828  13m  3916 S          1
0.2   0:46.35   /usr/local/apache2/bin/httpd
```

How to view the individual CPUs in the top command?

While the top command is running, press 1 (number one), which will display the performance data of the individual CPUs on that machine as shown below.

```
top - 13:10:13 up 171 days, 20:21,  3 users,  load
average: 0.01, 0.05, 0.00
Tasks: 194 total,   1 running, 193 sleeping,   0 stopped,
0 zombie
Cpu0  : 10.2% us,  2.6% sy,  0.0% ni, 86.8% id,  0.3% wa,
0.0% hi,  0.0% si
Cpu1  :  9.6% us,  8.0% sy,  0.0% ni, 82.4% id,  0.0% wa,
0.0% hi,  0.0% si
Cpu2  :  1.3% us,  1.3% sy,  0.0% ni, 95.0% id,  2.3% wa,
0.0% hi,  0.0% si
```

```
Cpu3  :  0.0% us,  0.0% sy,  0.0% ni, 100.0% id,  0.0% wa,
0.0% hi,  0.0% si
Mem:   1034624k total,  1007420k used,       27204k free,
174540k buffers
Swap:  2031608k total,               0k used,  2031608k
free,    615904k cached
```

Additional Top Command Examples:

15 Practical Linux Top Command Examples

Top on Steroids – 15 Practical Linux HTOP Examples

How To Capture Unix Top Command Output to a File in Readable Format

IFTOP Guide: Display Network Interface Bandwidth Usage on Linux

Hack 91. Df Command

df command (disk free) displays the amount of total and free disk space available on the mounted filesystems.

```
Syntax: df [options] [name]
```

How much GB of disk space is free on my system?

Use df -h as shown below. Option -h displays the values in human readable format (for example: K for Kb, M for Mb and G for Gb). In the sample output below, / filesystem has 17GB of disk space available and /home/user filesystem has 70GB available.

```
# df –h
Filesystem            Size  Used Avail Use% Mounted on
```

```
/dev/sda1              64G   44G   17G  73%   /
/dev/sdb1             137G   67G   70G  49%   /home/user
```

What type of filesystem do I have on my system?

Option -T will display the information about the filesystem Type. In this example / and /home/user filesystems are ext2. Option -a will display all the filesystems, including the 0 size special filesystem used by the system.

```
# df -Tha
Filesystem    Type     Size  Used  Avail Use%  Mounted on
/dev/sda1     ext2     64G    44G   17G  73%    /
/dev/sdb1     ext2    137G    67G   70G  49%    /home/user
none          proc       0      0     0    -    /proc
none          sysfs      0      0     0    -    /sys
none          devpts     0      0     0    -    /dev/pts
none          tmpfs   2.0G      0  2.0G    0%   /dev/shm
```

Hack 92. Du Command

du command (disk usage) will print the file space usage for a particular directory and its subdirectories.

How much space is taken by my home directory and all its subdirectories?

In the following example, option -s stands for summary only. i.e it displays only the total size of /home/jsmith and not the individual sizes of all the subdirectories inside the /home/jsmith. Option -h displays the information in a human readable format. i.e K for KB, M for MB and G for GB. The ~ indicates the user home directory. This command is same as “du -sh /home/jsmith”

```
# du -sh ~
320M    /home/jsmith
```

To get the subdirectories under /home/jsmith listed, execute the above command without the s option.

Hack 93. Lsof Commands

Lsof stands for ls open files, which will list all the open files in the system. The open files include network connection, devices and directories. The output of the lsof command will have the following columns:

- COMMAND process name.
- PID process ID
- USER Username
- FD file descriptor
- TYPE node type of the file
- DEVICE device number
- SIZE file size
- NODE node number
- NAME full path of the file name.

View all open files of the system

Execute the lsof command without any parameter as shown below.

```
# lsof | more
COMMAND PID  USER   FD   TYPE  DEVICE  SIZE   NODE NAME
init      1  root  cwd   DIR     8,1   4096      2 /
init      1  root  rtd   DIR     8,1   4096      2 /
init      1  root  txt   REG     8,1   32684  983101
/sbin/init
init      1  root  mem   REG     8,1   106397 166798
/lib/ld-2.3.4.so
init      1  root  mem   REG     8,1   1454802 166799
/lib/tls/libc-2.3.4.so
```

```
init      1  root  mem   REG     8,1   53736   163964
/lib/libsepol.so.1
init      1  root  mem   REG     8,1   56328   166811
/lib/libselinux.so.1
skipped…
```

The lsof command by itself without may return lot of records as output, which may not be very meaningful except to give you a rough idea about how many files are open in the system at any given point of view as shown below.

```
# lsof | wc -l
3093
```

View open files by a specific user

Use lsof –u option to display all the files opened by a specific user.

```
# lsof –u ramesh
vi      7190 ramesh  txt    REG        8,1   474608
475196 /bin/vi
sshd    7163 ramesh    3u  IPv6   15088263
TCP dev-db:ssh->abc-12-12-12-12.socal.res.rr.com:2631
(ESTABLISHED)
```

A system administrator can use this command to get some idea on what users are executing on the system.

List Users of a particular file

If you like to view all the users who are using a particular file, use lsof as shown below. In this example, it displays all users who are currently using vi.

```
# lsof /bin/vi
COMMAND  PID  USER  FD   TYPE DEVICE   SIZE   NODE NAME
vi      7258 root   txt   REG    8,1 474608 475196 /bin/vi
```

```
vi      7300 ramesh txt    REG    8,1 474608 475196 /bin/vi
```

Hack 94. Vmstat Command

For a typical performance monitoring all you need is only vmstat command. This display memory, swap, IO, system and cpu performance information.

The following command executes vmstat every 1 second for 100 times.

```
# vmstat 1 100
procs -----------memory---------- ---swap-- -----io----
--system-- ----cpu----
 r  b   swpd   free   buff  cache   si   so    bi    bo
in    cs us sy id wa
 0  0      0 282120 134108 5797012    0    0     0     2
0     0  0  0 100  0
 0  0      0 282120 134108 5797012    0    0     0     0
1007   359  0  0 100  0
 0  0      0 282120 134108 5797012    0    0     0     0
1117   577  0  0 100  0
 0  0      0 282120 134108 5797012    0    0     0     0
1007   366  0  0 100  0
```

Vmstat procs Section

- r field: Total number of runnable process
- b field: Total number of blocked process

Memory section

- Swpd field: Used swap space
- Free field: Available free RAM
- Buff field: RAM used for buffers
- Cache field: RAM used for filesystem cache

Swap Section

- Si field: Amount of memory swapped from disk per second
- So field: Amount of memory swapped to disk per second

IO Section

- Bi field: Blocks received from disk
- Bo field: Blocks sent to disk.

System Section

- In field: Number of interrupts per second.
- Cs field: Number of context switches per second.

CPU Section

- Us field: Time spend running user code. (non-kernel code)
- Sy field: Time spent running kernel code.
- Id field: Idle time.
- Wa field: Time spent waiting for the IO

Additional vmstat Examples:

24 iostat, vmstat and mpstat Examples for Linux Performance Monitoring

Hack 95. Netstat Command

Netstat command displays the network related information such as network connections, routing tables, interface statistics. Following are few examples on how to use netstat command.

Display Active Internet Connections and domain sockets using netstat

```
# netstat –an


Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address Foreign Address   State
tcp        0      0 0.0.0.0:5666  0.0.0.0:*        LISTEN
tcp        0      0 0.0.0.0:111   0.0.0.0:*        LISTEN
tcp        0      0 0.0.0.0:4086  0.0.0.0:*        LISTEN

skipped..

Active UNIX domain sockets (servers and established)
Proto RefCnt Flags       Type       State         I-Node
Path
unix  2      [ ACC ]     STREAM     LISTENING     7894
/tmp/.font-unix/fs7100
unix  2      [ ACC ]     STREAM     LISTENING     9662
/tmp/.gdm_socket
unix  2      [ ACC ]     STREAM     LISTENING     10897
@/tmp/fam-root-
```

Display Active Connections with Process ID and Program Name

This could be very helpful to identify which program has initiated a specific network connection.

```
# netstat -tap


Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address  Foreign Address
State       PID/Program name
tcp        0      0 *:nrpe         *:*
LISTEN      16277/xinetd
tcp        0      0 localhost.localdomain:smtp  *:*
LISTEN      7263/sendmail: acce
```

```
tcp       34      0 localhost.localdomain:54221
localhost.localdomain:4089  CLOSE_WAIT  29881/httpd
tcp        0   3216 dev-db:ssh                 cpe-76-94-
215-154.soca:4682 ESTABLISHED 11717/sshd: ramesh
```

Display Routing Table

```
# netstat –-route
Kernel IP routing table
Destination Gateway     Genmask       Flags MSS irtt Iface
192.168.1.0 *            255.255.255.0 U      0   0   eth0
162.244.0.0 *            255.255.0.0   U      0   0   eth0
default     192.168.1.1  0.0.0.0       UG     0   0   eth0
```

Display RAW network statistics

```
# netstat --statistics --raw
Ip:
    11080343 total packets received
    0 forwarded
    1 with unknown protocol
    0 incoming packets discarded
    11037744 incoming packets delivered
    11199763 requests sent out
Icmp:
    577135 ICMP messages received
    64 input ICMP message failed.
    ICMP input histogram:
        destination unreachable: 537
        timeout in transit: 65
        source quenches: 2
        echo requests: 576476
        echo replies: 12
```

```
        timestamp request: 3
        address mask request: 3
    581558 ICMP messages sent
    0 ICMP messages failed
    ICMP output histogram:
        destination unreachable: 5079
        echo replies: 576476
        timestamp replies: 3
```

Misc Netstat Commands

- # netstat –-tcp –-numeric List of TCP connection to and from the machine.
- # netstat --tcp --listening –-programs Display TCP port that the server is listening on along with the program that is listening on that particular port.
- # netstat –rnC Display the routing cache

Additional Netstat Examples:

10 Netstat Command Examples

Hack 96. Sysctl Command

Linux kernel parameter can be changed on the fly using sysctl command. Sysctl helps to configure the Linux kernel parameters during runtime.

```
# sysctl –a
dev.cdrom.autoclose = 1
fs.quota.writes = 0
kernel.ctrl-alt-del = 0
kernel.domainname = (none)
```

```
kernel.exec-shield = 1
net.core.somaxconn = 128
net.ipv4.tcp_window_scaling = 1
net.ipv4.tcp_wmem = 4096        16384   131072
net.ipv6.route.mtu_expires = 600
sunrpc.udp_slot_table_entries = 16
vm.block_dump = 0
```

Modify Kernel parameter in /etc/sysctl.conf for permanent change

After modifying the kernel parameter in the /etc/sysctl.conf, execute sysctl –p to commit the changes. The changes will still be there after the reboot.

```
# vi /etc/sysctl.conf


# sysctl –p
```

Modify kernel parameter temporarily

To temporarily modify a kernel parameter, execute the following command. Please note that after reboot these changes will be lost.

```
# sysctl –w {variable-name=value}
```

Hack 97. Nice Command

Kernel decides how much processor time is required for a process based on the nice value. Possible nice value range is: -20 to 20. A process that has a nice value of -20 is very high priority. The process that has a nice value of 20 is very low priority.

Use ps axl to display the nice value of all running process as shown below.

```
# ps axl
```

```
F   UID   PID  PPID PRI  NI   VSZ  RSS WCHAN  STAT TTY
TIME COMMAND
4     0     1     0  16   0  2172  552 -      S    ?
0:17 init [5]
1     0     3     1  34  19     0    0 ksofti SN   ?
3:18 [ksoftirqd/0]
1     0    10     1   5 -10     0    0 worker S<   ?
0:01 [events/0]
4     0  5145     1  25  10 32124 18592 -      SNs  ?
0:08 /usr/bin/python /usr/bin/rhn-applet-gui --sm-client-
id default4
4     0  5147  5142  16   0  3528  604 -      S    ?
0:00 /sbin/pam_timestamp_check -d root
1   503 17552  4180  16   0 14208 3920 -      S    ?
0:01 /home/www/apache2/bin/httpd -f
/home/www/apache2/conf/httpd.conf -k start
```

How to assign a low priority to a shell-script? (higher nice value)

In the example below, when I started the nice-test.sh script in the background, it took the nice value of 0.

```
$ ./nice-test.sh &
[3] 13009


$ ps axl | grep nice-test
0   509 13009 12863  17   0  4652  972 wait   S    pts/1
0:00 /bin/bash ./nice-test.sh


[Note: 6th column with value 0 is the nice.]
```

Now, let us execute the same shell script with a different nice value as shown below.

```
$ nice -10 ./nice-test.sh &
[1] 13016
```

```
$ ps axl | grep nice-test
0   509 13016 12863  30  10  4236  968 wait   SN   pts/1
0:00 /bin/bash ./nice-test.sh
```

[**Note**: 6th column with value 10 is the nice value for the shell-script.]

How to assign a high priority to a shell-script? (Lower nice value)

In the following example, let us assign a nice value of -10 (minus 10) to the nice-test.sh shellscript.

```
$ nice --10 ./nice-test.sh &
[1] 13021
$ nice: cannot set priority: Permission denied
```

Note: Only root user can set a negative nice value. Login as root and try the same. Please note that there is a double dash before the 10 in the nice command below.

```
# nice --10 ./nice-test.sh &
[1] 13060

# ps axl | grep nice-test
4     0 13060 13024  10 -10  5388  964 wait   S<   pts/1
0:00 /bin/bash ./nice-test.sh
```

[**Note**: 6th column with value -10 is the nice value of the shell-script.]

Hack 98. Renice Command

Renice alters the scheduling priority of a running process.

How to decrease the priority of a running process? (Increase nice)

In the example below, an existing shell-script is running at nice value of 10. (6th column in the ps output)

```
 $ ps axl | grep nice-test
0   509 13245 13216  30  10  5244  968 wait   SN   pts/1
0:00 /bin/bash ./nice-test.sh
```

To increase the nice value (thus reducing the priority), execute the renice command as shown below.

```
$ renice 16 -p 13245
13245: old priority 10, new priority 16


$ ps axl | grep nice-test
0   509 13245 13216  36  16  5244  968 wait   SN   pts/1
0:00 /bin/bash ./nice-test.sh


[Note: Now, the 6th column of the nice-test.sh (PID 13245)
shows the new nice value of 16.]
```

How to increase the priority of a running process? (Decrease nice)

In the example below, an existing shell-script is running at a nice value of 10. (6th column in the ps output)

```
$ ps axl | grep nice-test
0   509 13254 13216  30  10  4412  968 wait   SN   pts/1
0:00 /bin/bash ./nice-test.sh
```

In increase the priority, give a lower nice value as shown below. However, only root can increase the priority of a running process, else you'll get the following error message.

```
$ renice 5 -p 13254
```

```
renice: 13254: setpriority: Permission denied
Login as root to increase the priority of a running
process


$ su -


# renice 5 -p 13254
13254: old priority 10, new priority 5


# ps axl | grep nice-test
0   509 13254 13216  25   5  4412  968 wait   SN   pts/1
0:00 /bin/bash ./nice-test.sh


[Note: The 6th column now shows a lower nice value of 5
(increased priority)]
```

Hack 99. Kill Command

kill command can be used to terminate a running process. Typically this command is used to kill processes that are hanging and not responding.

```
Syntax: kill [options] [pids|commands]
```

How to kill a hanging process?

First, identify the process id of the particular process that you would like to kill using the ps command. Once you know the process id, pass it as a parameter to the kill command. The example below shows how to kill the hanging apache httpd process. Please note that typically you should use “apachectl stop” to stop apache.

```
# ps aux | grep httpd
USER       PID %CPU %MEM    VSZ    RSS TTY  STAT START
TIME COMMAND
apache   31186       0.0        1.6  23736 17556 ?
S          Jul26       0:40  /usr/local/apache2/bin/httpd
```

```
apache   31187       0.0        1.3  20640 14444 ?
S          Jul26       0:37  /usr/local/apache2/bin/httpd


# kill 31186 31187
```

Please note that the above command tries to terminate the process graciously by sending a signal called SIGTERM. If the process does not get terminated, you can forcefully terminate the process by passing a signal called SIGKILL, using the option -9 as shown below. You should either be the owner of the process or a privileged user to kill a process.

```
# kill -9 31186 31187
```

Another way to kill multiple processes easily is by adding the following two functions to the .bash_profile.

```
function psgrep ()
{
 ps aux | grep "$1" | grep -v 'grep'
}


function psterm ()
{
 [ ${#} -eq 0 ] && echo "usage: $FUNCNAME STRING" &&
return 0
 local pid
 pid=$(ps ax | grep "$1" | grep -v grep | awk '{ print
$1 }')
 echo -e "terminating '$1' / process(es):\n$pid"
 kill -SIGTERM $pid
}
```

Now do the following, to identify and kill all httpd processes.

```
# psgrep http
```

```
USER       PID %CPU %MEM    VSZ    RSS TTY  STAT START
TIME COMMAND
apache   31186       0.0        1.6  23736 17556 ?
S          Jul26       0:40  /usr/local/apache2/bin/httpd
apache   31187       0.0        1.3  20640 14444 ?
S          Jul26       0:37  /usr/local/apache2/bin/httpd


# psterm httpd
terminating 'httpd' / process(es):
31186
31187
```

Additional Kill Examples:

4 Ways to Kill a Process – kill, killall, pkill, xkill

Hack 100. Ps Command

ps command (process status) will display snapshot information of all active processes.

```
Syntax: ps [options]
```

How to display all the processes running in the system?

Use "ps aux", as shown below.

```
# ps aux | more
USER       PID %CPU %MEM    VSZ    RSS TTY  STAT START
TIME COMMAND
root         1       0.0    0.0    2044     588 ?   Ss
Jun27      0:00  init [5]
apache   31186       0.0    1.6  23736 17556 ?        S
Jul26       0:40  /usr/local/apache2/bin/httpd
```

```
apache   31187       0.0    1.3  20640 14444 ?        S
Jul26       0:37  /usr/local/apache2/bin/httpd
```

You can also use "ps -ef | more", to get a similar output

Print the Process Tree

You can use either ps axuf or ps –ejH to display processes in a tree format. The tree structure will help to visualize the process and it's parent process immediately. For clarity purpose, few columns have been cut-off in the output below.

```
# ps axuf
root     Oct14   0:00 /opt/VRTSralus/bin/beremote
root     Oct14   0:00  \_ /opt/VRTSralus/bin/beremote
root     Oct14   0:00      \_ /opt/VRTSralus/bin/beremote
root     Oct14   0:00      \_ /opt/VRTSralus/bin/beremote
root     Oct14   0:01      \_ /opt/VRTSralus/bin/beremote
root     Oct 14  0:00      \_ /opt/VRTSralus/bin/beremote
root     Dec03   0:01 /usr/local/sbin/sshd
root     Dec22   1:08 /usr/local/sbin/sshd
root     23:35   0:00  \_ /usr/local/sbin/sshd
511      23:35   0:00      \_ -bash
511                           \_ ps axuf
```

Note: You can also use pstree command to display process in tree structure.

View Processes Owned by a Particular User

The following command displays all the process owned by Linux username: oracle.

```
$ ps U oracle
PID TTY      STAT   TIME COMMAND
```

```
5014 ?        Ss     0:01 /oracle/bin/tnslsnr
  7124 ?        Ss     0:00 ora_q002_med
8206 ?        Ss     0:00 ora_cjq0_med
8852 ?        Ss     0:01 ora_pmon_med
8854 ?        Ss     0:00 ora_psp0_med
8911 ?        Ss     0:02 oraclemed (LOCAL=NO)
```

View Processes Owned by Current User

Following command displays all the process owned by the current user.

```
$ ps U $USER
PID TTY      STAT   TIME COMMAND
10329 ?        S      0:00 sshd: ramesh@pts/1,pts/2
10330 pts/1    Ss     0:00 -bash
10354 pts/2    Ss+    0:00 -bash
10530 pts/1    R+     0:00 ps U ramesh
```

Additional PS Examples:

7 Practical PS Command Examples for Process Monitoring

Hack 101. Sar Command

Sar commands comes with the sysstat package. Make sure sysstat is installed. If you don't have sar installed on your system, get it from Sysstat project.

Sar is an excellent monitoring tool that displays performance data of pretty much every resource of the system including CPU, memory, IO, paging, networking, interrupts etc.,

Sar Collects, Reports (displays) and Saves the performance data. Let us look at all the three aspects separately

Sadc - System activity data collector

/usr/lib/sadc (System activity data collector) command collects the system data at a specified time interval. This uses the daily activity data file that is located under /va/log/sa/sa[dd], where dd is the current day.

Sa1 shell-script

/usr/lib/sa1 in-turn calls the /usr/lib/sadcs. sa1 is invoked from the crontab as shown below. Run this every 5 minutes or 15 minutes depending on your need. I prefer to schedule it for every 5 minutes in the cron tab as shown below.

```
*/5 * * * * root /usr/lib/sa/sa1 1 1
```

Sa2 shell-script

/usr/lib/sa2 is a shell script that will write a daily report in the /var/log/sa/sa[dd] file, where dd is the current day. Invoke the sa2 from the crontab once a day at midnight.

```
# 59 23 * * * root /usr/lib/sa/sa2 –A
```

Note: /etc/cron.d/sysstat files comes with the sysstat package that includes some default value for the sa1 and sa2, which you can change accordingly.

Display CPU Statistics using Sar Command

```
# sar –u
Linux 2.6.9-42.ELsmp (dev-db)        01/01/2009
12:00:01 AM  CPU  %user  %nice   %system   %iowait
%idle
12:05:01 AM  all  3.70    0.00      0.85      0.00
95.45
```

```
12:10:01 AM  all  4.59    0.00      1.19      0.06
94.16
12:15:01 AM  all  3.90    0.00      0.95      0.04
95.11
12:20:01 AM  all  4.06    0.00      1.00      0.01
94.93
12:25:01 AM  all  3.89    0.00      0.87      0.00
95.23
12:30:01 AM  all  3.89    0.00      0.87      0.00
95.23


Skipped..


Average: all     4.56     0.00      1.00      0.15
94.29
```

Note: If you need a break down of the performance data for the individual CPU's, execute the following command.

```
# sar -u -P ALL
```

Display Disk IO Statistics using sar command

```
# sar –d
Linux 2.6.9-42.ELsmp (dev-db)        01/01/2009
12:00:01 AM    DEV              tps    rd_sec/s  wr_sec/s
12:05:01 AM    dev2-0           1.65      1.28     45.43
12:10:01 AM    dev8-1          4.08      8.11     21.81


Skipped..


Average:       dev2-0           4.66    120.77     69.45
Average:       dev8-1          1.89      3.17      8.02
```

Display networking Statistics using sar command

```
# sar -n DEV | more
Linux 2.6.9-42.ELsmp (dev-db)        01/01/2009
12:00:01 AM     IFACE   rxpck/s   txpck/s   rxbyt/s
txbyt/s   rxcmp/s   txcmp/ s  rxmcst/s
12:05:01 AM        lo      0.17      0.16     25.31
23.33      0.00      0.0 0      0.00
12:10:01 AM      eth0     52.92     53.64  10169.74
12178.57      0.00      0.00      0.00


# sar -n SOCK |more
Linux 2.6.9-42.ELsmp (dev-db)        01/01/2009
12:00:01 AM  totsck   tcpsck    udpsck    rawsck   ip-frag
12:05:01 AM      50       13         3         0         0
12:10:01 AM      50       13         4         0         0
12:15:01 AM      53       13         5         0         0
```

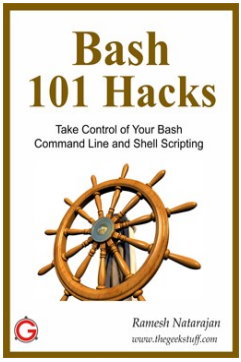
Additional SAR Examples:

10 Useful Sar (Sysstat) Examples for Linux Performance Monitoring

Your Support is Appreciated

If you liked this Linux 101 hacks free ebook, and would like to thank me and support The Geek Stuff blog, purchase the following eBooks:

Bash 101 Hacks

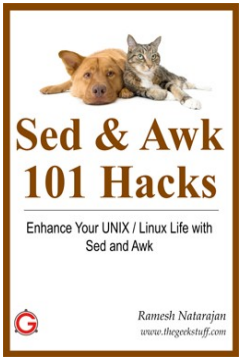


Bash is the default shell on Linux. If you are spending lot of time on Linux environment, you should master the Bash command line features to become efficient. Apart from being an interactive shell, Bash is also a scripting language, which allows you to automate your tasks using Bash shell scripting.

Bash 101 Hacks is a downloadable eBook that contains 101 practical examples on both Bash command line and shell scripting.

Get Your Copy of: Bash 101 Hacks eBook

Sed and Awk 101 Hacks

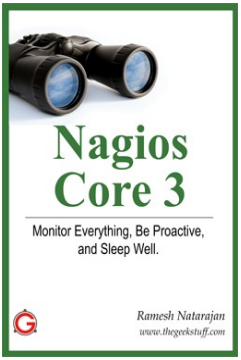


If you are spending lot of time on UNIX / Linux, you'll be manipulating text files frequently. You may be making the similar edits on multiple configuration files on one or more servers. You may be digging huge log files (or data files) looking for certain information.

Sed and Awk 101 Hacks is a downloadable eBook that contains 101 practical examples on various advanced Sed and Awk features, that will help you understand everything you need to know about Sed and Awk.

Get Your Copy of: Sed and Awk 101 Hacks eBook

Nagios Core 3

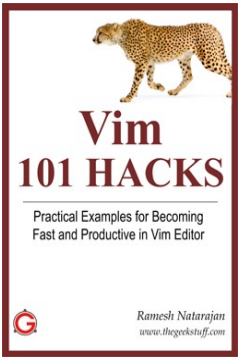


You should implement a robust monitoring solution that will notify you when there is an issue. It should also notify the right people at the right time about a potential issue, even before it becomes critical.

Nagios Core 3 eBook is the only guide you'll ever need to get your IT infrastructure monitored using Nagios Core, and it will help you to understand everything you need to know to implement Nagios Core 3.

Get Your Copy of: Nagios Core 3 eBook

Vim 101 Hacks



Vim editor is very powerful editor that will make you extremely productive once you take some time to learn and understand its features. If you are like most people, you would like to have a structured way of learning this powerful editor and take advantage of all its features.

Vim 101 Hacks is a downloadable eBook that contains 101 practical examples on various advanced Vim features that will make you fast and productive in the Vim editor.

Get Your Copy of: Vim 101 Hacks eBook

10 Amazing and Essential Linux Books

For further reading on Linux, I recommend the following books. The 10 Linux books mentioned here by no means are comprehensive or authoritative list. But, these 10 Books are few of my favorites that I enjoyed reading over the years and I strongly believe will enhance your technical abilities on Linux, if you have not read them yet.

1. **SSH, The Secure Shell, by Daniel J. Barrett, Richard E. Silverman and Robert G. Byrnes**. This is hands-down the best book on SSH. This book explains both theoretical and practical aspects of SSH. Using SSH as an end-user is fairly straight forward . But, configuring SSH as an administrator is complex and involves a detailed understanding of SSH. This is a must read for any system administrator. The examples in this book show exactly what needs to be done differently for the different flavors of SSH such as SSH1, SSH2 and OpenSSH.
2. **Essential System Administration, by Æleen Frisch**. This is an excellent book for those who like to become a Unix System Administrator. This book covers all the typical system administration tasks. This is a perfect companion when you are dealing with multiple flavors of Unix, as it has examples for AIX, FreeBSD, HP-UX, Linux, Solaris and Tru64. I've used the pocket version of this book — Essential System Administration Pocket Reference, when I was managing multiple flavors of Unix systems at the same time.
3. **Linux Server Hacks, Volume One, by Rob Flickenger**. 100 awesome practical hacks packed in one book. Setup a Linux test bed and try out all these hacks. These hacks are neatly grouped into different sections — Server Basics, Revision Control, Backups, Networking, Monitoring, SSH, Scripting, and Information Servers. Once you've mastered these hacks, you should absolutely read Linux Server Hacks, Volume Two, by William von Hagen and Brian Jones, which has 100 Linux hacks focused on authentication, monitoring, security, performance and connectivity.
4. **DNS and BIND, by Cricket Liu and Paul Albitz**. Several years ago, I configured my first DNS by reading online documentation. I brought this book to understand how DNS

and BIND works. I've already upgraded this book twice when a newer edition was released. This should definitely be in your library, if you are a serious system administrator.

5. **Understanding the Linux Kernel, by Daniel Bovet and Marco Cesati.** If you are a serious developer on Linux environment or a sysadmin, this is a must read. This books explains the inner workings of the Linux Kernel 2.6 in a structured and logical way. This talks about how Kenel handles the Memory Management, Process scheduling, I/O architecture and Block devices. Overall this book is a treat for geeks who are curious to explore what is under the hood of Linux.
6. **The AWK Programming Language, by Alfred V. Aho, Brian W. Kernighan, Peter J. Weinberger.** This is a classic book written written by the authors of Awk. If you are dealing with text based data files on Linux environment, mastering Awk will help you to quickly create scripts to automate your data file manipulation jobs.
7. **Linux Cookbook, by Carla Schroder.** This book covers Linux features from both users and system administrators point of view. There are two chapters dedicated for installing and managing software on RPM-based system and Debian. If you use RedHat, the Linux Pocket Guide, by Daniel J. Barrett is an excellent addition to your library, which covers all the essential Linux command with a sample usage.
8. **Linux Firewalls, by Michael Rash.** To build a secure Linux system, you must read this book. There are quiet few books out there for iptables. But, this one talks specifically about the fundamentals of how to configure an Intrusion Detection System using iptables, psad and fwsnort. If you want a comprehensive handy reference of all the things iptables can do with specific examples, Linux Iptables Pocket Reference, by Gregor N. Purdy is the best.
9. **Linux Administration Handbook, by Evi Nemeth, Garth Snyder and Trent R. Hein.** During my early days of system administration, I've referred this book frequently. This is pretty detailed book with close to 1000 pages and 30 chapters that are nicely grouped together in three high level sections — Basic Administration, Networking and Bunch O' Stuff.
10. **Beginning Ubuntu Linux, by Keir Thomas and Jaime Sicam.** For those who like to transition from Windows to

Linux, install Ubuntu Linux on one of your old laptop or desktop and get this book. I strongly believe in spreading the news about Linux to those who don't use it. If you want any of your loved ones or friends to learn Linux, install Ubuntu on an old laptop and give this book as a gift to them. They'll definitely be very thankful to you.

More Recommended Books:

For Additional Linux and Open Source Related Books that I recommend, visit The Geek Stuff Book Store at Amazon

Extended Reading

Following are few articles from the The Geek Stuff blog for your extended reading. Check out The Geek Stuff Archives section for more articles.

1. 50 Linux Sysadmin Tutorials
2. 50 Most Frequently Used Linux Commands (With Examples)
3. Unix LS Command: 15 Practical Examples
4. Turbocharge PuTTY with 12 Powerful Add-Ons
5. wget Tutorial: 15 Awesome Examples to Download Files from Internet
6. Ping Tutorial: 15 Effective Ping Command Examples
7. Nagios - Enterprise Monitoring Solution
   - Nagios Jumpstart Guide
   - Monitor Window Server
   - Monitor Linux Server
   - Monitor Network Switch
   - Monitor VPN Device
8. Perform SSH and SCP without entering password:
   - From openSSH to openSSH
   - From openSSH to SSH2
   - From SSH2 to SSH2
9. Hello World Examples (Learn a programming language)
10. UNIX Sed Tips and Tricks
11. Ubuntu Tips and Tricks
12. MySQL Tutorials
13. PostgreSQL Tutorials
14. Vi / Vim Tips and Tricks
    - Vim Macro Tutorial: How To Record and Play
    - How To Use Vim as Perl IDE and C/C++ IDE
    - Automatic Word Completion in Vim

15. 3 Steps to Add Custom Header to a File Using Vim
16. The Ultimate Guide for Creating Strong Passwords
17. 6 Steps to Secure Your Home Wireless Network
18. Firefox Add-On: Hire 7 Personal Bodyguards to Browse Internet Securely
19. Tripwire Tutorial: Linux Host Based Intrusion Detection System
20. Midnight Commander (mc) Guide: Powerful Text based File Manager for Unix

More Linux Articles

I publish Linux and Open Source articles on an on-going basis on The Geek Stuff blog.

To get Linux Tips, HowTos, Guides and Tutorials on an on-going basis, subscribe to The Geek Stuff blog. If you subscribe, you will get new articles posted on TGS website directly to your inbox or to your RSS reader.

Subscribe to receive free Linux tutorials directly to your email inbox regularly.

Thank You

I hope you found the ***Linux 101 Hacks*** eBook helpful.

I sincerely appreciate all the support given by you and other regular readers of my thegeekstuff.com blog.

You have encouraged me in more ways than you know.

If you liked this eBook, and would like to support me, consider buying my other eBooks:

- Bash 101 Hacks
- Sed and Awk 101 Hacks
- Vim 101 Hacks
- Nagios Core 3

If you have any suggestions, or feedback, or questions while reading this ebook, don't hesitate to reach out to me. You can connect with me on the following:

- Twitter (@thegeekstuff)
- Facebook page

If you want to write to me directly, you can use this contact form to reach out to me.

Ramesh Natarajan

ramesh@thegeekstuff.com